hEYEbrid: A hybrid approach for mobile calibration-free gaze estimation

CHRISTIAN LANDER, German Research Center for Artificial Intelligence (DFKI GmbH), Saarland Informatics Campus, Germany
MARKUS LÖCHTEFELD, Aalborg University, Denmark
ANTONIO KRÜGER, German Research Center for Artificial Intelligence (DFKI GmbH), Saarland Informatics Campus, Germany

We introduce *hEYEbrid*, a calibration-free method for spontaneous and long-term eye gaze tracking, with competitive gaze estimation. It is based on a hybrid concept that combines infrared eye images with corneal imaging. For this, two eye cameras are mounted on a glasses frame. In this way, the pupil can be tracked quickly with high precision. This information is translated into the corneal image, which is used to create a connection to the environment, acting like a scene camera. In a user study with 20 participants, we evaluated our approach against an extended version of the system, called *3C-hEYEbrid*, and a state-of-the-art head-mounted Pupil Labs eye tracker. We show that *hEYEbrid* provides accurate gaze estimation in unconstrained environments and is robust against calibration drift (e.g. caused by taking off and putting on the device). In addition, we present a mobile and wearable implementation of *hEYEbrid* and *3C-hEYEbrid*, that is also usable with a monocular Pupil Labs eye tracker. It connects the head-mounted device to a mobile phone, enabling gaze estimation in real time. *hEYEbrid* represents a significant step towards pervasive gaze-based interfaces.

CCS Concepts: • **Human-centered computing** → **Interaction devices**; **Ubiquitous and mobile devices**; • **Computing methodologies** → *Computer vision*;

Additional Key Words and Phrases: Corneal Imaging, Eye Tracking, Gaze Estimation, accuracy

ACM Reference Format:
Christian Lander, Markus Löchtefeld, and Antonio Krüger. 2017. hEYEbrid: A hybrid approach for mobile calibration-free gaze estimation. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 4, Article 149 (December 2017), 29 pages. https://doi.org/10.1145/3161166

1 INTRODUCTION

The visual system allows us to perceive a highly detailed reflection of our environment and plays an important role when interacting with the real world. The point of gaze reveals where we are looking, and is used to quantify our attention and interests [71]. Hence, eye gaze has been used as an input modality in human computer interaction since the early 90s [19]. Recently, gaze-based interfaces in stationary settings became a valid option, as remote eye trackers became easily available at a low price point. The latest devices for usage with desktop computers are available for under $200 (Tobii 4C [69]). However, making gaze-based interfaces ubiquitous remains an

Authors' addresses: Christian Lander, German Research Center for Artificial Intelligence (DFKI GmbH), Saarland Informatics Campus, Stuhlsatzenhausweg 3, Saarbrücken, 66123, Germany; Markus Löchtefeld, Aalborg University, Rendsburggade 14, Aalborg, 9000, Denmark; Antonio Krüger, German Research Center for Artificial Intelligence (DFKI GmbH), Saarland Informatics Campus, Stuhlsatzenhausweg 3, Saarbrücken, 66123, Germany.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2017 Association for Computing Machinery.
2474-9567/2017/12-ART149 $15.00
https://doi.org/10.1145/3161166

open challenge that cannot be resolved using remote technologies. In particular, the integration of gaze support in next-gen AR and VR devices seems inevitable, as the latest industrial evolutions suggest [64, 65]. For the remainder of the article we will focus on head-mounted eye trackers, as these are the most promising technology for pervasive gaze-based interfaces [3].
These eye trackers consist of a glasses frame that is equipped with at least two cameras: first, an eye camera capturing a close-up image of the user's eye, and second, a world camera partially recording the user's current field of view. The purpose of the eye camera is to detect and track the pupil as well as its movements. A widely used approach to achieve this is active illumination of the eye with infrared. Gaze estimation is the process of mapping the pupil positions from the eye camera's into the world camera's coordinate system. A crucial step towards gaze estimation is calibration, which is used to create a function that maps eye to gaze positions. Current state-of-the-art eye trackers, such as Tobii Pro Glasses 2 [66], are built on model-based (geometric) gaze estimation. This method enables calibration-less gaze estimation. To adapt the model and compute personal parameters like the angle kappa (compare [33]), at least a one- dpoint calibration is needed. To further increase the accuracy to be competitive, even more calibration points are needed [72]. Further, these methods need complex hardware setups with more than two cameras and are extremely expensive (more than tens of thousands of USD). On the other hand's, eye trackers that rely on regression-based (interpolation) gaze estimation need a more extensive calibration step, but are usually cheaper [24].
All gaze estimation methods share the same set of parameters, that are acquired through the calibration. According to Hansen et al. [14], these are:

- *camera calibration* to determine the intrinsic camera parameters
- *geometric calibration* to determine relative locations and orientations in the setup, like camera and light sources
- *personal calibration* to estimate the cornea curvature, the angle kappa (offset between visual and pupillary axes)
- *gaze mapping calibration* to determine the parameters of the eye to gaze mapping function

Even if it is possible to set up a fully calibrated eye tracker, that is, a system in which the camera parameters and the geometry are known, the so-called calibration drift will affect the gaze estimation accuracy. This effect can be caused by several factors (e.g. eye physiology, environmental factors, the device's position) and is known to worsen the gaze estimation [26]. Besides that, the current appropriate calibration procedures usually require the user to fixate a sequence of visual stimuli. That is, a supervisor, usually a professional, is needed to check the validity of the data sampling, the positioning of the cameras and finally the accuracy of the recorded calibration. To summarize, this step itself constitutes a serious problem that prevents the application of current head-mounted eye trackers at pervasive scale [32]. Desirable attributes of an eye tracker include minimal intrusiveness and obstruction, allowing for free movements while maintaining high accuracy, easy and flexible setup, and low cost. A more detailed description of eye tracker preferences is given by Scott and Findlay [53]. Although the latest commercial eye trackers try to include all these attributes, they remain regrettably expensive. Moreover, most of these devices rely on a direct connection to a powerful computer for real-time data processing. As a result, current head-mounted eye trackers are neither applicable for spontaneous interaction nor suitable to conduct long-lasting in-the-wild experiments.

In this paper we present *hEYEbrid*, a concept that enables calibration-free and accurate gaze estimation using a head-mounted device. We developed a hybrid approach to estimate a person's gaze, utilizing two eye cameras: (1) an infrared eye camera that is tracking the pupil's positions and its movements, and (2) a natural light eye camera that captures closeup RGB images of the eye. The latter captures the environment reflected on the corneal surface, i.e. the iris and the pupil of the human eye, and is referred to as the corneal camera in the rest

of the paper. The underlying assumption is that the pupil center in the corneal camera's images corresponds to the actual gaze point in the real environment. This assumption is similar to the one that most current eye-trackers make. Pupil position and center are extracted in the infrared image and mapped onto the corneal image. Hence we are able to extract the reflection within the pupil area, which roughly represents the user's current field of view. Both cameras are placed next to each other in a fixed position, focusing on the same eye. With *hEYEbrid*, we do eliminate the need for a cumbersome calibration as well as regular re-calibration procedures. To compute the relationship between the infrared and corneal camera image plane, a mapping needs to be computed at one time during the construction process of the head-mounted device. Thus *hEYEbrid* is an enabling technology for any kind of gaze estimation in pervasive settings, such as spontaneous gaze estimation or user studies in the wild.

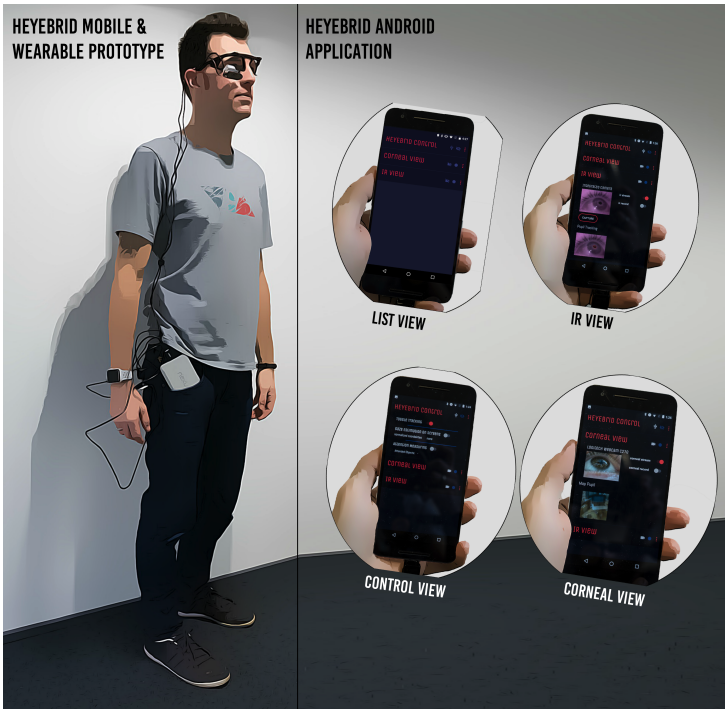

Fig. 1. The developed *hEYEbrid* concept used to build a mobile and wearable device for spontaneous competitive gaze estimation at any place and any time. The head-mounted device is connected to a Nexus 6P that processes both camera streams in real time.

In a controlled user study with 20 participants we evaluated the gaze estimation accuracy of three approaches, each different in the number and location of cameras: (1) *hEYEbrid* using two eye cameras, (2) *3Camera hEYEbrid* (*3C-hEYEbrid*), which combines the two eye cameras of *hEYEbrid* with an additional scene camera, and (3) a state-of-the-art monocular *Pupil Labs* eye tracking system[1]. In three validation sessions, each participant had to focus on 15 different objects. Between sessions, we either re-calibrated the Pupil Labs eye tracker or asked the participant to take off and put on the device to simulate a calibration drift. With *hEYEbrid*, we achieved the highest gaze estimation accuracy, compared to the other approaches, namely 2.19°. The simulated calibration drift

[1]https://pupil-labs.com/

had no significant impact on *hEYEbrid*'s performance, as it still achieved 2.36° accuracy in a fully unconstrained setting.

Finally, we designed and implemented a mobile and wearable system by connecting a head-mounted device based on *hEYEbrid* to an Android phone, as illustrated in figure 1. It includes the functionalities for ad-hoc gaze estimation on a public screen and focused object detection. Moreover, the mobile application is able to drive a head-mounted Pupil Labs eye tracker as well as the *3C-hEYEbrid* device.

In sum, our work provides the following contributions:

- **Novel hybrid approach** connecting IR eye and corneal images.
- **Accurate and constant long-term** gaze estimation without (re-)calibration.
- **Wearable and mobile** system to employ gaze as an input modality in fully pervasive settings.

The remainder of the paper is structured as follows: We first give an in-depth description of the existing eye tracking and gaze estimation techniques and their problems. We also review the existing work tackling the problem of calibration. After that, we introduce the concept *hEYEbrid* as well as its extension, *3C-hEYEbrid*, and describe the implementation. We then describe the conducted study and present its results. In the last part of the paper, we illustrate the design and implementation of the mobile prototype.

2 BACKGROUND & RELATED WORK

In this section we will first provide an overview about the anatomy of the human eye and the parts that are important for eye tracking and gaze estimation. After that we will focus on the description and presentation of different eye tracking and gaze estimation methods, as well as their current problems.

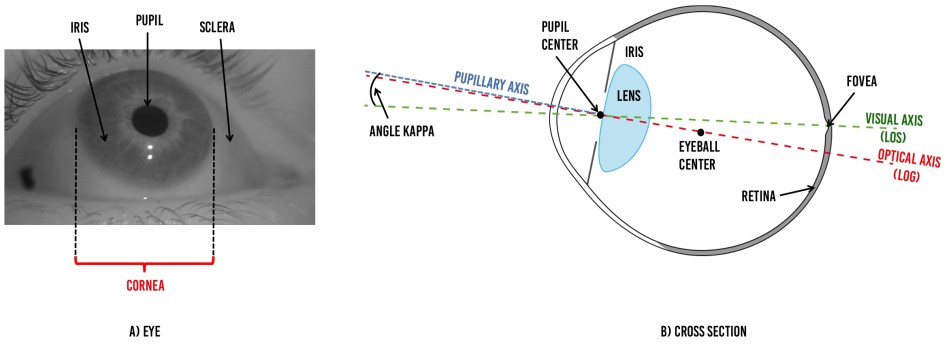

Fig. 2. Snapshots of the human eye. a) shows a picture facing the human eye; b) shows a cross section of the eye.

2.1 The Eye

The human eye is our organ of vision. The eyeball is made up of three layers, as shown in figure 2a). The *cornea* is a clear transparent convex layer at the front of the eye; the white part is the *sclera*, which is a continuation of the cornea and protects the sensitive mechanism of the organ. The third and innermost layer, the *retina*, contains the rods and cones, which are the light-sensitive receptors, connected to the optic nerve. The fovea centralis (often fovea) is the region of the most accurate vision, which constitutes about 5° of vision. Between cornea and lens, the iris is located, whose function is to to adapt the pupil diameter according to different light conditions. The pupil is a black circular opening in the center of the iris, through which light is passed to the lens and the retina. The lens is a biconvex body refracting (bending) the incoming light, so that it can be focused on the retina. The border or edge between the cornea and sclera is called the limbus [7].

Most important for gaze estimation is the definition of the axes of the eye. In a simplified model, there are two main axes, as depicted in Figure 2b). The optical axis is defined by the line connecting the center of curvatures of the eye, i.e. the center of the eyeball, the pupil and the lens. It is often referred to as the line of gaze (LoG). The visual axis is defined as the line passing through the fovea and the center of the pupil. It is often referred to as the line of sight (LoS) and determines the real gaze direction. The pupillary axis is the normal line to the corneal surface passing through the center of the pupil, which is slightly different from the optical axis. Most important for gaze estimation is the so-called angle kappa that is formed between the line of sight and the pupillary axis. A definition of all axes and their respective angles can be found in [33].
The movements performed by the human eye are divided into several categories [16]. Basically, one distinguishes between *fixations*, *saccades* and *smooth-pursuit* movements [14]. Fixation is not a real movement by itself, as it describes the state of resting the gaze at a specific target within the central vision for a certain amount of time, typically around 200–300 milliseconds (ms). Saccades are fast, jump-like movements of the eye to re-focus it between two consecutive fixations, with a duration of 30–80 ms. Smooth pursuit movements are performed, when following an object that moves with a constant speed (e.g., gazing at a passing car). The movements, our eyes perform throughout the day are linked to cognitive processes and can be used to analyze people's visual behavior [56].

2.2 Eye Tracking & Gaze Estimation

Nowadays, three main techniques exist to track a person's eye and do gaze estimation. The electro-oculography (EOG) based method places electrodes around the eye to measure the skin potentials, as proposed by Kaufman et al. [23]. The scleral search coil method uses a small coil that is embedded into a contact lens. It measures the voltage caused by an external electro-magnetic field [49]. Camera-based (video-based) techniques use the images of the eye to detect its characteristics in combination with images of the field of view to realize gaze estimation. This method is mainly applied nowadays, by either using a head-mounted (cameras are mounted on the head) or a remote (cameras are placed in the environment) system. As *hEYEbrid* belongs to the category of head-mounted video-based eye gaze trackers, we will focus on these techniques. For a detailed review of eye gaze tracking methods, we refer the reader to Young and Sheena [73]. In the following section we will provide the reader with an overview of video-based eye tracking and gaze estimation methods. Although it is not the main focus of the article, it is important to understand the current problems and how we try to tackle them with our proposed method.

Detecting and tracking the eye is an essential step towards gaze estimation. Eye detection and tracking is still a challenging task, as there are many unforeseen issues that have to be taken into account, like occlusion of the eye by the eyelids, degree of openness of the eye, variability in size, reflectivity or head pose, etc. The eye can be characterized through different features including the intensity distribution of the pupil, iris, and cornea, as well as by their shapes. Ethnicity, viewing angle, head pose, color, texture, light conditions, the position of the iris within the eye socket, and the state of the eye (i.e., open/closed) are issues that heavily influence the appearance of the eye. According to Hansen et al. [14], there exists a huge taxonomy of eye detection techniques, consisting of *shape-based*, *appearance-based* and *hybrid* methods. To get an in-depth review and comparison of different approaches for each method, we refer the reader to them. We will provide a briefer description of the approaches in the following.
Shape-based approaches are based on a pre-defined geometric model of the eye's shape that is used to match against. Existing methods differ in the level of detail, ranging from simple elliptical to complex models. Feature-based shape methods detect the eye according to its features, such as edges, eye corners, the limbus (border between iris and sclera) or pupil. The appearance-based (holistic) approach is comparable with a template matching. For this an image patch model is created and eye detection is done using a similarity measure [75].

The difference from other methods is that appearance-based techniques detect the eyes directly based on the photometric appearance (color, filter response). Existing hybrid methods simply combine the aforementioned methods to exploit their respective benefits.
The most prominent eye detection method nowadays is to use active infrared (IR) illumination [14]. This is primarily done to enhance the contrast between pupil and iris. It is a special technique that cannot be mapped exclusively to one of the groups from above. We distinguish between methods relying on visible light, called *passive*, and invisible light, called *active* approaches. Most active methods use near IR light sources (780-880 nm), invisible for the human eye, thus not disturbing the user. Depending on the location of the light source with respect to the camera, the pupil appears to be bright (close to the optical axis of the camera) [36] or dark (away from the camera's optical axis). The most popular algorithm is Starburst [30], introduced in 2015. It is a video-based eye detection method, using the pupil contours as feature point, that are mapped onto an ellipse shape using RANSAC model fitting. Recently Fuhl et al. [11] compared the latest pupil detection algorithms ElSe [10], ExCuSe [9], Labs [22], SET [20], Swirski [60] and Starburst using different datasets of active illuminated dark-pupil images. All these methods are feature-based methods using ellipse models and work on IR eye images. They found that all algorithms give reasonable results. However, when it comes to real world images revealing problems like changing illumination, occlusions, or reflections [52], the ElSe algorithm outperforms existing approaches with respect to detection rate.
In summary, the recent development of more robust and accurate pupil detection algorithms is a major step towards pervasive head-mounted eye tracking, which is a requirement for mobile gaze estimation.

Gaze estimation is the primary task of gaze trackers, which are often just referred to as eye trackers. Gaze can be defined as either the gaze direction or the point of regard (PoR). A person's gaze is usually determined by the head pose (position and orientation) together with the eyeball rotation. The gaze changes if at least one of the two values varies. It is common that a person first brings the head into a comfortable position before orientating the eyes. To achieve very accurate gaze estimation, head movements have to be included. This is done by tracking it via additional hardware (e.g., tracking the head, as done with remote trackers), or integrating them directly into the method used.
Different methods exist to realize gaze estimation, which is always about finding a suitable mapping from pupil to gaze positions. Depending on the hardware approach, pupil positions are mapped into the scene video of a head-mounted device, or onto the screen if using a remote approach. The gaze estimation method by itself is independent of the hardware approach. The majority use feature-based methods, which are divided into two subclasses. *Geometric* (or model-based) ones calculate a 3D gaze direction vector by modeling the eye as a sphere. The center of the cornea is estimated, and thus the optical axis can be computed. Such methods rely on prior knowledge of personal eye parameters, such as size and radii of the eyeball and the cornea. Recently, Swirski et al. [61] proposed a new ellipse fitting model that automatically constructs the 3D eye model. They were able to achieve an average gaze error of 1.68°. However, they used perfect simulated noiseless data to provide ground-truth results. In general, model-based approaches may require specific hardware and 3D knowledge about the scene that is not always available, especially in mobile settings.
Regression (or interpolation-based) methods typically create a 2D mapping from pupil to gaze positions. Various approaches exist to assess this mapping by polynomial functions[4], neural networks [21] or homography transformations [74]. Although regression-based gaze estimation is able to achieve highly accurate results (up to 0.5°, as reported in [32]), it usually requires a user-dependent calibration to evaluate the parameters needed for the mapping. These include information about the relationship between the cameras as well as personal parameters like the angle kappa, using the Purkinje images [8].
An alternative approach for gaze estimation is taken by *appearance-based* methods that analyze the pixel intensity of the raw eye image. These methods work without a user calibration, but provide only a rough accuracy. For

instance, Zhang et al. [75] report on 6.1° on average.
In summary, all present gaze estimation methods require some kind of calibration process [14]; they differ solely in the type and the amount of calibration, which influences the gaze estimation accuracy. With *hEYEbrid* we aim to eliminate the need for a calibration while still enabling reasonable gaze estimation in mobile settings. In the following, we will highlight the problems of calibration and current investigations to address these.

2.3 Calibration

According to Holmqvist et al. [16], accuracy is one major characteristics of data quality in eye trackers. It is defined as the distance between the estimated gaze position of the device and the actual gaze location in the environment (where the person is really looking). Inaccuracy is directly influenced by the eye tracker itself; that can be a result of bad pupil detection and tracking, a poor mapping from pupil to gaze positions, or both. As we presented earlier, much research has been done recently on improving the pupil detection, including under realistic conditions [10]. Thus the source of error is more related to a bad pupil-to-gaze mapping, which is a direct result of a poor calibration. The calibration is performed by looking at a number of pre-defined visual stimuli. While the user is fixating the calibration target, data is sampled that consists of pupil positions in the eye image, the physical orientation of the eye and the location of the target on the calibration plane (i.e. for remote trackers the screen, for head-mounted devices the scene camera images) [13]. But the calibration procedure itself leads to many problems.
First of all, the calibration is an individual process that has to be conducted for each user separately prior to usage [8, 16]. The reasoning is that the calibration includes human-specific parameters (e.g., cornea curvature and angle kappa) and geometric information (e.g., relative location and orientation between cameras) [14].
Nyström et al. [45] investigated calibration with respect to practical issues. Typically a second person is needed to set up and calibrate the eye tracking device. They compared different approaches of setting up the device and sampling the calibration data. With an operator approach, a second person is doing all the work, whereas in a participant-controlled approach, the users themselves monitor the data sampling. A system-controlled approach does an automatic collection of calibration data, as the decision whether or not the user is currently fixating the calibration target is an automated process. Nyström et al. [45] found that a participant-controlled approach yields the best results. Automatic decision by the system still achieves better accuracy and precision than an operator-controlled calibration. It is noteworthy that at least for remote eye tracking devices, the major manufacturers use an automatic calibration (e.g., Tobii 4C[2] [68]). However, modern high-end head-mounted eye trackers such as the Tobii Pro Glasses 2 [67] still require the support of an additional person acting as operator. Also, the developer-friendly Labs eye tracker applies a combination of the approaches depending on the use case [24]. Although this might not be a huge problem per se, the occurrence of the so-called calibration drift requires a regular re-calibration to maintain accurate gaze estimation. This effect describes the deterioration of gaze estimation accuracy due to various reasons. These include changes in eye physiology (e.g., wetness of the eye), the environment (e.g., lighting conditions), the position of the device and/or the camera in relation to the eye, and changes of the user's location and orientation [5]. Necessary re-calibration in the sense of executing the full procedure to replace outdated calibration data is thereby time-consuming and distracts from the actual task. Existing methods use a sampling subset to reduce time and effort [26, 55] and keep track of the user's position in space [25].
In sum, the user, location and orientation dependency, the calibration drift and the need for supervision prevent head-mounted eye gaze trackers from being used outside of a controlled environment in pervasive settings. Existing research addresses the aforementioned issues partially, but there exists no work so far that resolves all of them. A prominent approach, for instance, is to make use of smooth pursuit movements of the eye. Pfeuffer et

[2]https://help.tobii.com/hc/en-us/articles/213414285-Specifications-for-4C

	remote techniques				head-mounted techniques				
	[39, 40]	[51]	[34]	[41]	[35]	[62]	[63]	[12]	*hEYEbrid*
average accuracy	N/A	15.14 mm to 25.13 mm	<1°		2.51°	N/A	9.5°	2.5°	2.19°
3D pose estimation	yes				yes				no
calibration	yes	no	yes	no	no	yes	no	yes	no
illumination	none	active display	multiple IR LEDs	none	2 LEDs	none	none	4 IR LEDs	IR LED
cameras	one	one biocular	two scene + eye	one	one	two monocular	one	two biocular	two monocular

Table 1. An overview of existing gaze estimation approaches based on corneal imaging. It shows the main differences between *hEYEbrid*and remote and head-mounted techniques.

al. [46] calibrate a remote eye tracker while following moving objects on a screen in an unobtrusive way. The movements of the eye are correlated with known trajectories of the moving objects (up to 0.6° of gaze estimation accuracy). In a similar manner, Huang et al. [17] record calibration data during interaction with a computer (2.56° error). However, both approaches are restricted to a specific interface, such as a display, and are developed for remote devices. Another approach is to make use of visual saliency maps to auto-calibrate the eye gaze tracker [6, 59]. Combining video and EOG-based devices with saliency maps computed on the scene videos enables one to (re)-calibrate a head-mounted eye tracker [58] (6° of error in the best case). Although these works reduce the calibration drift through an automatic calibration, they create other issues elsewhere, such as cumbersome setups (cameras plus EOG) and inaccuracy. Recently Santini et al. [50] developed an approach for unsupervised calibration of a head-mounted eye tracker. For this, they turn the user's mobile phone into a calibration target that is moved around to recalibrate the system while still reaching highly accurate results (0.59° of gaze estimation error). But the approach is limited, as the recalibration has to be triggered manually. Hence the user has to know that the accuracy is getting worse due to calibration drift.

With *hEYEbrid* we make use of the mirror-like characteristics of the eye's cornea and exploit the fact, that the environment is reflected on the human eye. By combining infrared images with corneal imaging in a head-mounted device, we allow for calibration-free gaze estimation in unconstrained environments.

2.4 Gaze Estimation using Corneal Imaging

Different research areas took advantage of the specular reflections on the eye. Nishino and Nayar [39, 40] pioneered corneal reflection analysis, giving an analysis of what information the reflected image of an eye reveals. They developed the corneal catadioptric imaging system using a geometric model of the cornea. The derived system can be used for several applications like facial reconstruction and relighting [38], face recognition [37] and the calibration of display-camera setups [42]. Besides applications in the field of computer vision [43], several approaches utilize the reflection of the eye to enable gaze estimation [44]. Nakazawa et al. [34] used infrared light to create patterns in the surroundings, visible in the reflection image, to map the user's gaze into the environment. Nitschke et al. [41] further improved the method by omitting the active illumination. However, these approaches limited the user's mobility by using a remote camera. They achieved highly accurate gaze

estimation (< 1°error), but in a highly constrained setting while sitting in front of a large screen. Nakazawa et al. [35] used a head-mounted device to capture corneal reflections and achieved an average gaze estimation accuracy between 2.51° and 4.65° in a highly constrained static desktop setting, not competitive with current commercial remote desktop eye tracking systems. Takemura et al. [62, 63] developed a mobile prototype to estimate the object a user is focusing on. They utilized natural feature tracking to detect objects visible in the corneal images. Both systems are based on 3D eye pose estimation and geometric modeling of the eye, making them computationally expensive. Their current implementations achieve only 7.3 and 1 fps respectively. In addition, these works lack a detailed evaluation of the gaze estimation accuracy in realistic settings. Their two-camera prototype uses a corneal and a scene camera, and thus relies on a user-specific calibration. Consequently, both cannot be used for gaze based interaction in pervasive settings. Recently El Hafi et al. [12] developed a mobile corneal imaging system based on high-resolution cameras (4k) capturing both eyes. They show the possibility to reliably detect objects in the corneal reflections. In their experiment they evaluated the gaze estimation in a desktop setting while looking at different targets on a computer screen and achieved an average error of 2.5°. However, their installed 4K cameras can only deliver images at 11 fps, slowing down the processing speed. Table 1 highlights the primary differences of *hEYEbrid* compared to the aforementioned approaches. For the sake of completeness, we also included the remote techniques. To summarize, *hEYEbrid* is advancing the existing head-mounted approaches as it (1) introduces a new concept, without a 3D pose estimation and limbus tracking; (2) is evaluated in a mobile setting and (3) achieves better gaze estimation accuracy than existing approaches (especially for on-screen gaze estimation, with an average of 1.64°).

With *hEYEbrid*, we propose a novel hybrid concept that enables spontaneous and accurate gaze estimation in real time. The head-mounted device can be connected to a mobile phone. The user can dynamically choose to use our system, without the need for calibration or the support of other persons.

3 HEYEBRID

As stated in the section above, gaze estimation using a head-mounted eye tracker faces several challenges that come along with the main problem of calibration. Ideally, the user should be able to put on a device that is immediately ready to use and provides robust and accurate gaze estimation. In keeping with this idea, we introduce the concept of *hEYEbrid* in the following.

3.1 Concept

The basic idea of the approach is that the pupil center in the corneal image, i.e. in the reflected scene, coincides with the actual gaze in the real environment. In Figure 3, the concept is visualized. The area within the corneal limbus (often only limbus), that is the pupil and the iris, reflects the scene a person is currently facing. As the eye is a spherical object, it acts like a fish-eye view into the world from the person's perspective. Hence the reflection is a distorted representation that is blended with the structure of the iris and is clearest within the area of the pupil. Figure 3 also highlights the two most important axes of the eye, when it comes to gaze estimation. Standard head-mounted eye tracking systems model the angle kappa, i.e. the difference between visual and pupillary axes, using the vector of the first Purkinje and the pupil center. Note that both axes are going through the pupil center, i.e., the pupil center on the corneal reflection refers to the gaze.

Corneal images cover the whole eye, including the complete visible eyeball, eyelashes and parts of the skin around the eye. As only the area within the corneal limbus reveals a reflection of the environment, the parts outside can be neglected. Existing works that use corneal reflections for gaze estimation are based on limbus tracking [35, 41, 62]. To accomplish an exact and robust extraction on the corneal images, these approaches rely

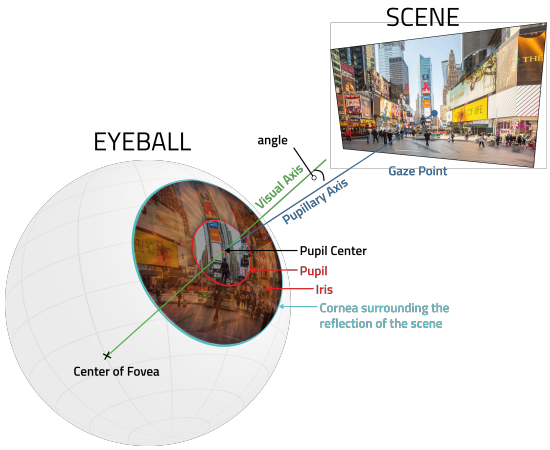

Fig. 3. Basic idea of *hEYEbrid* to use the pupil center in the corneal image (i.e. the environment reflected on the human eye) as the gaze point in the actual scene.

on a combination of feature- and model-based methods. Although the limbus can be distinguished well from the surrounding white sclera, other image features, such as eyelashes and eyelids, make the separation harder. In addition, the underlying models are complex to compute [44, 62, 63]. That is why we decided to use a different approach in *hEYEbrid*.

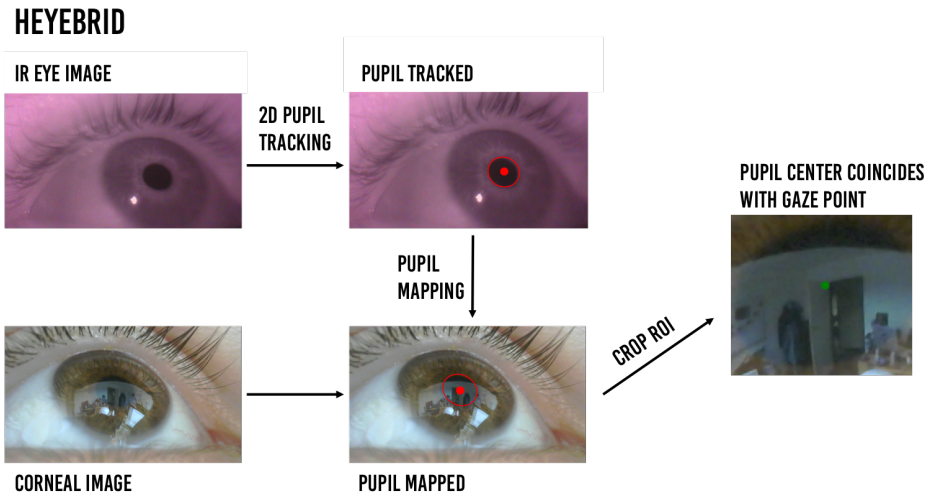

Fig. 4. Basic processing pipeline of *hEYEbrid*, which combines infrared eye and corneal images. The pupil is tracked in the IR image and mapped onto the corneal image to finally crop the reflection within the pupil. The mapped pupil center coincides with the gaze point.

Figure 4 illustrates the processing pipeline which we propose. Our concept is based on pupil detection and tracking. The advantages are that the pupil naturally reflects our current field of view and the reflection within

this area is explicit (unlike the iris' reflectance). *hEYEbrid* is based on a two-camera approach, as it is hardly possible to extract the pupil in the corneal image itself. Two different types of eye images – an infrared eye and a corneal image – are combined and processed simultaneously. In this way we can profit from the matured algorithms enabling a fast and robust pupil detection/tracking based on IR images. In each IR frame we obtain an ellipse describing the pupil's structure and center. This is mapped onto the corneal image, to finally crop the reflection around the pupil, as depicted in Figure 4.

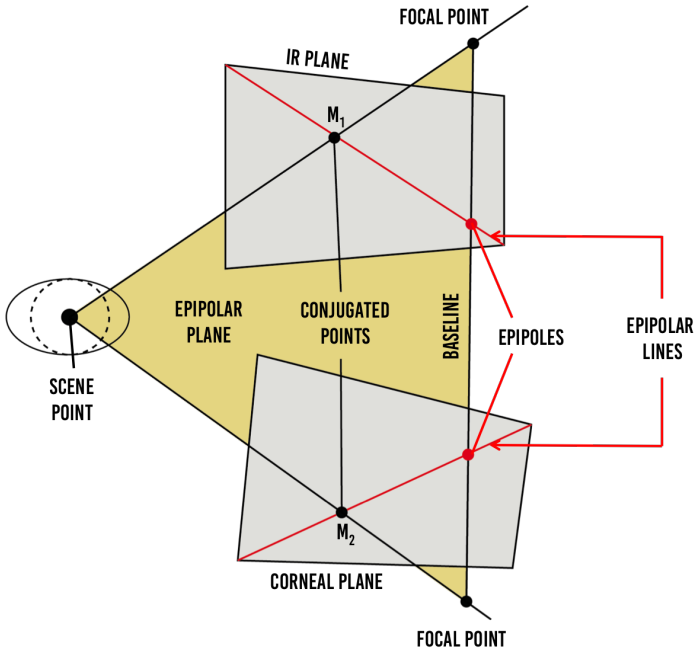

Fig. 5. Estimating a mapping between the two eye camera image planes based on epipolar geometry. In *hEYEbrid*, the problem is reduced from 3D to 2D and solved via a planar homography mapping matrix.

To project information from the infrared to the corneal image, it is necessary to obtain a mapping between the two image planes. Both cameras, the infrared as well as the corneal imaging camera, are placed in front of the same eye in a fixed position relative to each other. The necessary mapping has to be computed only once. We do this during the construction process of the head-mounted device. This mapping is a 3D transformation problem. Figure 5 visualizes the geometric solution of how to find the pupil, which we extracted in the IR plane, in the corneal plane.

In our system, we have two converging cameras in a specific position and orientation. According to the *Epipolar Constraint* [15] , the following equation holds:

$$m_2^T F m_1 = 0 \text{ with a suitable } 3 \times 3 \text{ matrix } \mathbf{F}, \text{ called a } \textit{fundamental matrix}$$

It says that a pixel m_2 of the corneal image plane, which corresponds to a pixel m_1 in the infrared image plane, cannot lie everywhere. The fundamental matrix has rank 2, thus is not invertible, and has seven degrees of freedom (DOF), i.e. 9 minus two for rank and scale. If the fundamental matrix is known, we know for each pixel

m_1 in the IR frame the corresponding epipolar line l_2 in the corneal frame and vice versa:

$$m_2^T l_2 = 0 \text{ with } l_2 = Fm_1$$
$$m_1^T l_1 = 0 \text{ with } l_1 = Fm_2$$

where the vector $l_i = (a, b, c)^T$ describes the epipolar line $ax + by + c = 0$. The fundamental matrix F can be computed from correspondences of image points. Given m_1, the search for m_2 is reduced to searching along the epipolar line.
In *hEYEbrid*, we simplify the camera transformation from 3D to a 2D problem. For this we assume the eye to be flat, so that the usual rotations of the pupil in 3D space are represented by a moving ellipse on a plane. That is the camera image plane, as it just records a 2D image. The original 3D information is encoded in changes of the elliptical parameters, such as minor and major axes, which result in different shapes of the pupil. In that case, the mapping reduces to searching for a transformation matrix between two planes. A planar *homography matrix* **H** satisfies the following constraint:

$$m_2 = Hm_1 \text{ for a } 3 \times 3 \text{ matrix } \mathbf{H} \text{ with rank } 3$$

There is a one-to-one point correspondence in this particular case, which can be computed using image features for instance. The homography matrix H is actually a projective transformation and has 8 degrees of freedom, 2 for scale, 2 for rotation, 2 for translation and 2 for line of infinity. It is important to note that it is not robust against changes in distance to the planes one wants to find the correspondence between. Consequently, we have to ensure that the distance from both cameras to the eye remains rather stable.
Finally, the combination of two eye cameras in a hybrid method is such that (1) we exactly compute the position of the pupil and its center without relying on 3D eye pose estimation using pre-defined parameters [35, 41] and (2) we do not require a user calibration [63]. The concept can be seen as just moving the scene camera of a video-based head-mounted eye tracker below the eye to capture the world through the eye of the user. In doing so, we eliminate the need for a calibration procedure.

3.2 Extension 3C-hEYEbrid

We modified *hEYEbrid* by adding scene images to the concept, based on the scene registration approach by Nakazawa et al. [2]. In Figure 6 the additional concept of *3C-hEYEbrid* is illustrated.

It projects the result of *hEYEbrid*'s pipeline, the cropped corneal image with the pupil center, to the current scene image. The mapping is done as in the method from above based on a planar projection. As we cannot guarantee that the spatial relationship between the corneal and scene camera is fixed, the correspondence is computed for each pair of corneal and scene frames. In particular, the cropped corneal image is used as a *template* image that we try to find in the *observed* image of the scene camera stream. We use a natural image feature approach that computes the homography matrix H if enough matches are found. The matrix describes a bidirectional mapping between the corneal and the scene camera's image plane. The pupil position that was already mapped on the corneal image is further transformed to gaze position in the scene image by applying the transformation matrix. This concept is similar to the standard approach of head-mounted eye trackers. But in contrast to performing a user calibration, *3C-hEYEbrid* is continuously doing a frame-based calibration to update the relationship between the cameras, without actively including the user. *3C-hEYEbrid* does not account for any image corrections (e.g., distortion) and does not rely on 3D eye pose estimation, like in [2]. Hence, it is more lightweight and faster.

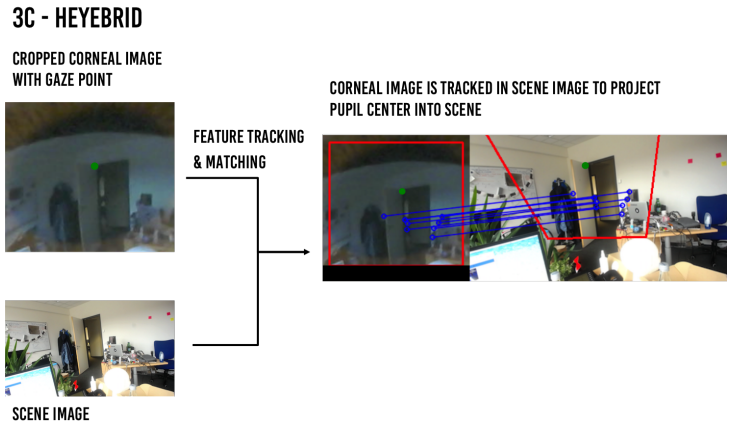

Fig. 6. The extension *3C-hEYEbrid* connects the result of *hEYEbrid*'s processing pipeline with the scene image to project the pupil center into the scene image plane.

3.3 Implementation

Both approaches are implemented using a monocular Pupil Labs Pro mobile eye tracking headset (2014 version[3]) of which the eye camera was extended by a corneal camera. Figure 7 illustrates the building steps to integrate *hEYEbrid* into the head-mounted device. We used a Logitech C270 webcam to capture corneal images with a maximal resolution of 1280 x 960 pixels at 30Hz. The original housing and the glue around the lens were removed to manually adjust the focus, so as to enable macro shots. This camera covers a 60° field of view and has a fixed focus of 4 mm. The cropped corneal images have a varying resolution after slicing the pupil region. Depending on the eye pose (i.e., the location of the pupil) the resolution ranges from 200 x 200 pixels to 300 x 300 pixels.

We reused the default infrared camera of the Pupil Labs eye tracker, but removed the housing. This camera can record images with a maximal resolution of 640 x 480 pixels at 30 Hz. We designed a new mount to position both cameras in front of the user's eye. It preserves the basic possibility to rotate the eye cameras and adds the option to be moved in front of the eye to position the cameras, so that the pupil is best covered. Both cameras are fixed with screws and placed at an angle of 150° to each other. Hence, it is guaranteed that the cameras do not move and can be placed close to the eye (around 25 mm), while still capturing the pupil and its movements.
The joint between the frame and the custom mount is lockable with a screw, so the mount and the cameras will not move without further effort. Consequently, the relationship between eye, cameras and scene is rather fixed and re-adjustments are done no more than for other head-mounted eye tracking devices. The custom mount was designed with Autodesk Fusion 360[4] and printed by a Formlabs Form2 3D printer[5] using transparent synthetic resin (GPCL02). All parts together (IR + corneal camera + custom mount) have a weight of 14 grams (compared to the 4 grams of the original Pupil Labs eye camera), and are therefore still comfortable to wear. The total weight is comparable to the default monocular device, as we removed the scene camera for *hEYEbrid*. The total size is $58 \times 28.25 \times 14.40mm$ (width x height x depth) compared to $46.20 \times 10.60 \times 7mm$ of the original eye camera. Hence it does not disturb the user's field of view any more than the usual device. For *3C-hEYEbrid*, we attached the

[3]https://pupil-labs.com/blog/2014-01/new-pupil-pro-headset-capture-software-0-3-7/
[4]http://www.autodesk.com/products/fusion-360/overview
[5]http://formlabs.com/3d-printers/form-2//

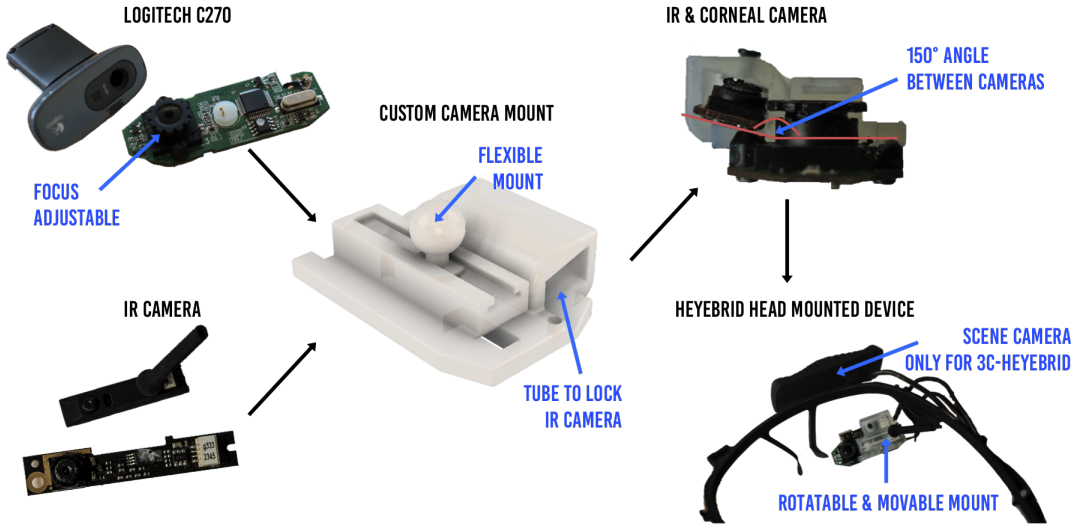

Fig. 7. Building steps to integrate *hEYEbrid* into the Pupil Labs head-mounted device. The corneal camera is affixed to a custom mount together with the device's IR camera. The mount is movable and rotatable.

device's default scene camera, a Logitech c920, capturing frames of 1920 x 1080 pixels at 30Hz. It has a 90° field of view with an auto-focus lens (shown in Figure 7) and a weight of about 80 grams.

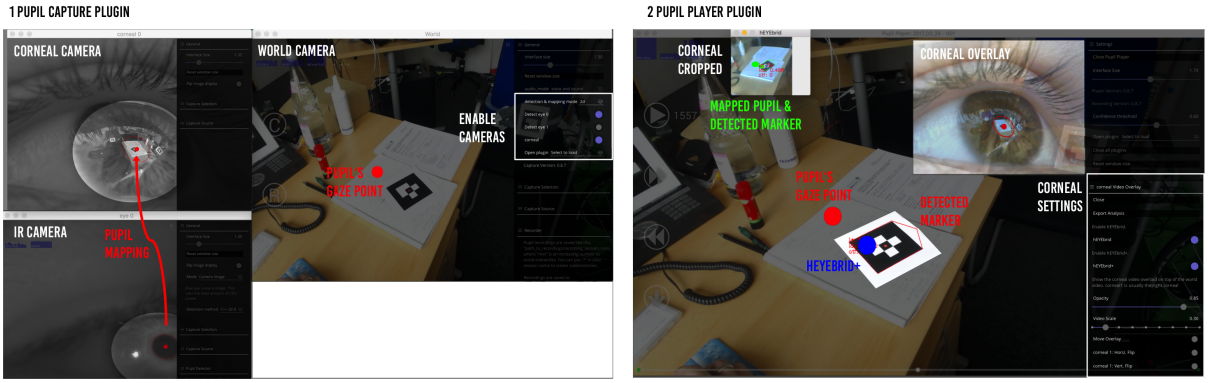

Fig. 8. Integration of *hEYEbrid* and *3C-hEYEbrid* into the PUPIL framework. The Capture software is able to record data, whereas the Player is used for analysis and gaze computation of the different approaches.

We integrated *hEYEbrid* and *3C-hEYEbrid* into the Pupil Labs open source eye tracking platform, version 0.8.7[6]. The modified devices are thus usable with laptops and desktop computers. We followed the best practices to extend the software[7] and implemented two new plugins for the Pupil Labs Capture and Player software, as

[6]http://github.com/pupil-labs/pupil/releases/tag/v0.8.7
[7]https://docs.pupil-labs.com/#plugin-guide

shown in Figure 8. The first extension we made is needed to integrate the new type of images from the corneal camera. It is similar to the existing plugin for IR images, in that it allows you to choose the camera source and to change its parameters (e.g., resolution, brightness, exposure time). The main purpose is to give direct feedback, i.e. the corneal image is overlaid with the pupil information, that is the ellipse and its center. We integrated the information about the corneal images into the software's interprocess communication. Hence it is possible to record the data of *hEYEbrid*, such as image data and coordinates of the mapped pupil, in conformity with Pupil Labs's logging and message format. The second extension is made to play back recorded data while using *hEYEbrid* or *3C-hEYEbrid*. We integrated additional functionalities to visualize the corneal images as well as their cropped version. All computed gaze points are highlighted in a different color depending on the approach. The built-in mechanism for AR marker tracking is also enabled for corneal images. For instance, it is possible to label and track objects. Finally, the user is able to export recorded data as CSV files for further processing and analysis.

We actively made use of all extensions for the analysis of the user study. All described software components are developed in Python v2.7.6. By default, the Pupil Labs software is able to manage camera streaming via pyuvc[8] (based on libuvc) and provides a toolkit for a graphical user interface through pyglui[9] (based on OpenGL). For image processing (e.g., feature tracking and matching, homography computation), methods of the OpenCV 3.2 library[10] are used. For simple image manipulations (e.g., flipping and slicing), we use numpy[11], as it provides fast array processing.

For pupil tracking, we used the software's built-in pupil detector[22] (implemented in detect_2d.hpp). It implements dark pupil tracking and is based on edge detection, contour tracking and ellipse fitting.

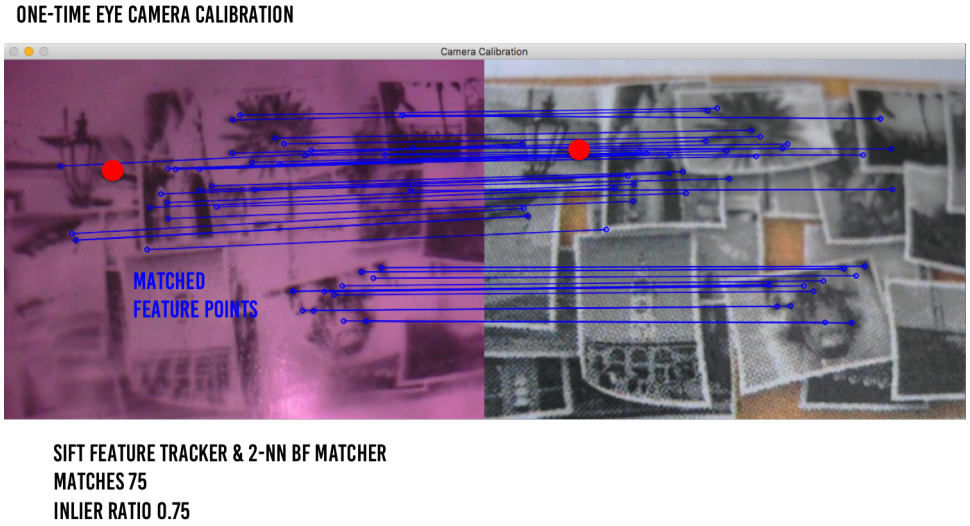

Fig. 9. Setting up the eye cameras by computing the transformation between the infrared and corneal image planes. Image features are extracted and matched to compute a homography matrix, used to map a point from one image into the other.

To map the pupil position onto the corneal images, the relation between both cameras was computed beforehand. This was done by estimating a homography matrix, needed for a bi-directional projective transformation between

[8]https://github.com/pupil-labs/pyuvc
[9]https://github.com/pupil-labs/pyglui
[10]http://opencv.org/
[11]http://www.numpy.org/

the IR and corneal image planes, as described before. Figure 9 depicts the one-time mapping computation. We captured images of the IR and corneal camera focusing on a printed image (size: $22 \times 15mm$), revealing lots of features. Both images are scaled to a resolution of 640×480 pixels. The homography matrix $\mathbf{H}$ is computed by first doing a feature detection using the scale-invariant SIFT algorithm[31]. In a second step, the image and the extracted features are matched with a 2-nn Brute-Force matcher. This simple matcher computes the distance between two features of two images, using the Hamming distance, and returns the two best matches. The resulting matches are used to find the homography matrix, by computing the perspective transformation, using the functions provided by OpenCV (cv2.findHomography()). Note that this procedure is only done once, during the construction process of the head-mounted device. As long as the cameras are fixed to each other, the procedure does not have to be repeated.
For *3C-hEYEbrid*, the Pupil Labs eye tracker's scene camera is used to retrieve the scene images. The homography matrix between the cropped corneal and scene image planes is computed as described above. We also use the methods for SIFT and a 2-nn Brute-Force matcher provided by OpenCV.
All extensions needed to integrate *hEYEbrid* and *3C-hEYEbrid* with the Pupil Labs framework will be made publicly available, together with the 3D models to build the hardware prototype.

4 EVALUATION

We conducted a controlled user study to assess the validity of *hEYEbrid*'s approach and its gaze estimation accuracy in comparison to an existing state-of-the-art eye tracking approach and *3C-hEYEbrid*. We collected data using the head-mounted device that was described above. With this it is possible to record data from a Pupil Labs eye tracker, *hEYEbrid*, and *3C-hEYEbrid* simultaneously and compare the gaze estimation accuracy.

4.1 Experimental Design

We used a *within-subject* 3 x 1 design with the independent variable *Mode*, i.e. the method used for gaze estimation. We chose the following three different modes for gaze computation: *hEYEbrid* and *3C-hEYEbrid* implemented as described above. In addition we used a state-of-the-art monocular *Pupil Labs* head-mounted eye tracking device (version 2014). We calibrated the Pupil Labs eye tracker for each participant separately. For this a 9-point marker calibration, standing in front of a 15-inch laptop screen, was performed using the built-in procedure of the Pupil Labs software. It is important to note that we did not record any calibration for *hEYEbrid* and *3C-hEYEbrid*, as both methods are calibration-free.

4.2 Task & Procedure

We implemented a gaze pointing task, in which each participant had to focus on 15 targets in total (10 tangible and 5 on screen), as shown in Figure 10. The instructor named an object (e.g., *book*), that the participant then had to look for. When the object was found, the participant had to focus on the center of a visual marker (indicated by a red dot) that was attached to the object. For on-screen targets, the participant had to look at a red circle every time the instructor named the target *projection*. Looking at the target was verbally acknowledged by the participant. Only then, the sampling was started, which lasted eight seconds for each target. During the task, the participant was able to freely move around. Figure 10 shows the schedule of the procedure. In the beginning of the experiment, we calibrated the Pupil Labs eye tracker. After this, each participant had to perform the task three times. Between the tasks half of the participants first took off and put back the head-mounted device (used to record each mode) and finally re-calibrated the Pupil Labs eye tracker. The other half of the participants did this the other way around, to counterbalance it.

We collected all the data, i.e. the raw video data from the infrared, corneal and scene camera, as well as information

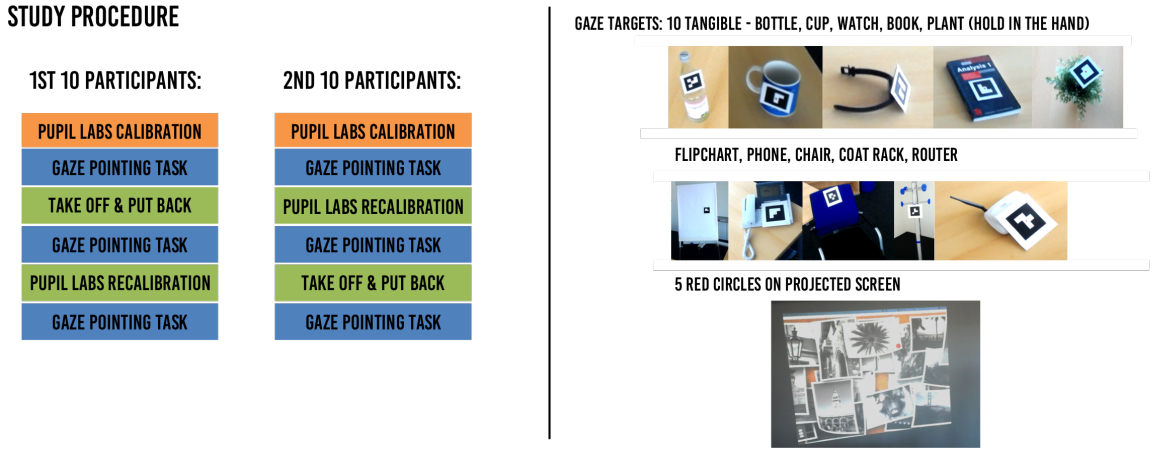

Fig. 10. The study protocol showing the procedure of the experiment as well as the gaze targets used for the gaze pointing task.

about the pupil tracking, its mapping and the computed gaze values for each approach in parallel at 30Hz. This was possible as we integrated all modes into one device. We captured 240 samples per object (8 seconds x 30 Hz), leading to a total of 3,600 samples per task (240 x 15 targets). As we only recorded after the acknowledgment of the user – when she started to fixate at the target center – we did not have to discard samples to account for the time that was needed to search for the object. Thus, we recorded 10,800 samples per participant (3,600 x 3 sessions), leading to a total number of 216,000 samples for each mode (*Pupil*, *hEYEbrid* and *3C-hEYEbrid*).

4.2.1 Apparatus. Figure 11 shows the study setup in a 360° picture. We set up a $4m^2$ (200 x 200 cm) square area, in which the participants were able to freely walk around. We did so to simulate a more realistic setting, in which people are moving around. Eight large tables (each 100 x 200 cm), forming four blocks of 200 x 200 cm, were arranged around this area. We used ten tangible objects, each different in size and form, of which seven were placed on the tables. The order of the objects' positions remained the same during the complete experiment. To track the objects and mark the gaze target, each object was equipped with an AR marker with a red dot at its center. Therefore, we printed ten AR markers – five of size 5 x 5cm and five 10 x 10 cm – and attached them onto the objects. The small markers indicated that the object had to be picked up by the user, using their hands. In addition we used a projected screen with a size of 210 x 160 cm (diagonal: 264 cm), that had a resolution of 1280 x 1024 pixels. Five additional targets were shown on the projected screen represented as red circles (diagonal: 40 px = 6.4 cm). The distribution of the on-screen targets is shown in Figure 11. Note that only one on-screen marker was shown at a time.

To validate the approach of *hEYEbrid*, we computed the gaze estimation accuracy and compared it against the values for *3C-hEYEbrid* and the Pupil Labs eye tracker. This eye tracker provides a gaze estimation accuracy of 0.6° with a precision of 0.08° [22]. Other prominent state-of-the-art devices, such as the SMI Eye Tracking Glasses 2, achieve a slightly better accuracy of 0.5° [54]. Since Pupil Labs provides an open source development framework, it perfectly fits the purpose of our research. To compare the results obtained by each approach, we computed the gaze estimation error in degrees of visual angle. This error describes the difference between the gaze target (indicated by the red dot of an AR marker or by a red on-screen circle) and the computed gaze point according to the mode. The smaller the error, the more accurate the mode. In the case of the Pupil Labs eye tracker and *3C-hEYEbrid*, we used the scene camera image as a basis to compute the distance between the gaze

Fig. 11. The study setup showing the distribution of the tangible objects (attached with AR markers) in the room and a participant gazing at the book. In addition the projection including all targets is shown.

point and the center location of the gaze target (AR marker or on-screen target). For *hEYEbrid*, we used the AR marker or on-screen target reflected on the corneal image and computed its distance to the pupil center. For marker detection and tracking, we used Pupil Labs's built-in marker tracking plugin that is based on ArUco[12]. For the analysis, i.e. the marker tracking and the computation of gaze accuracy, we used the player software including the custom plugins, as described above. We were able to detect the markers without problems in the scene and the corneal images. To convert the raw pixel distances to the degree of visual angle, the pixels per degree have to be estimated. For *Pupil* and *3C-hEYEbrid*, we used the field of view of the scene camera (90°). For *hEYEbrid*, we use 30° for the field of view of the pupil, as reported in [2, 35].

4.3 Participants

In total 20 participants (5 female) between 23 and 38 years old (M=28.8 years, SD=3.31) participated in the data collection. All participants were recruited from a local university campus and had normal or corrected-to-normal vision; none reported any form of visual impairments (e.g., color blindness).

4.4 Results

We first computed the overall gaze estimation error for all three modes across all tasks and all objects. Using *hEYEbrid* gave the lowest error (M=2.19°, SD=0.92°), followed by *Pupil* (M=3.09°, SD=2.26°), and *3C-hEYEbrid* (M=4.05°, SD=1.04°). We performed a one-way ANOVA on the gaze estimation accuracy across all three modes and found a significant difference ($F(2,17) = 7422.22$, $p = 0.000$). To further assess this finding, we computed three paired independent-samples t-tests on gaze estimation error. We found a significant difference in gaze estimation error between *hEYEbrid* and *Pupil* ($t(48)$=-4.11, p<0.001). No significant difference was found between *3C-hEYEbrid* and *Pupil* ($t(48)$=-0.45, p=0.65) and *hEYEbrid* and *3C-hEYEbrid* ($t(48)$=-0.87, p=0.38).

We subsequently analyzed the gaze estimation error for the different target groups – on-screen and tangible objects. The lowest error was achieved for the on-screen targets using the *hEYEbrid* mode. Table 1 lists these results for each mode.

Independent-samples t-tests were conducted and we found the same results for the tangible target group as for the overall results. *hEYEbrid* and *Pupil* differed significantly according to gaze estimation error ($t(48)$=-3.06,

[12]http://www.pupil-labs.com/blog/2013/12/036-release.html

	hEYEbrid		*3C-hEYEbrid*		Pupil	
	M	*SD*	*M*	*SD*	*M*	*SD*
tangible	2.22°	0.91°	4.26°	1.12°	3.27°	2.22°
on-screen	1.64°	0.28°	2.17°	1.21°	2.84°	1.38°

Table 2. Means and standard deviations for the computed gaze estimation errors for all modes on the two target groups.

p<0.05), as well as *hEYEbrid* and *3C-hEYEbrid* ($t(48)$=-1.3, p<0.05). No significant difference was found between *3C-hEYEbrid* and *Pupil*. For the on-screen markers, we were able to measure a significant difference in gaze estimation error between *hEYEbrid* and *Pupil* ($t(48)$=-2.93, p<0.05). We found no significant difference between *3C-hEYEbrid* and *Pupil* or *hEYEbrid* and *3C-hEYEbrid*.

We were further interested in the impact of the calibration of the *Pupil* eye tracker. We compared the gaze estimation accuracy of all three modes for each task separately. The tasks differ in the procedure that was done before doing the gaze pointing task (shown in Figure 10). Figure 12 summarizes the results. It shows the mean gaze estimation error for all three modes after the initial calibration of the *Pupil* eye tracker (shown in the first bar group). In the second group the mean gaze error after a re-calibration of the *Pupil* eye tracker is shown. In the third group the mean gaze estimation error of the different modes after a simulated calibration drift, i.e

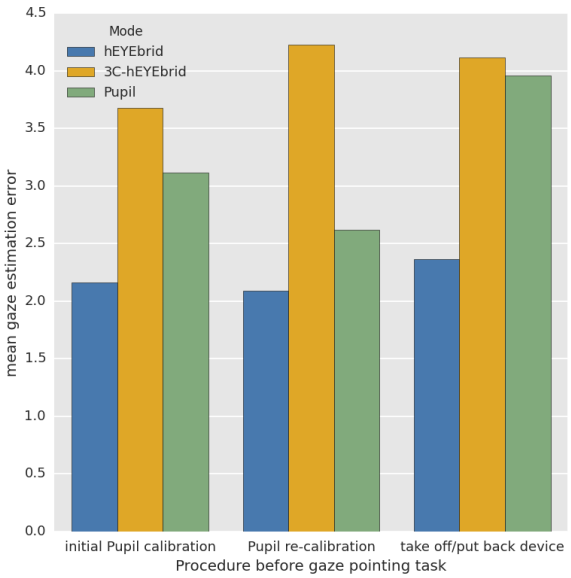

Fig. 12. Mean gaze estimation error for each mode for each of the three gaze pointing tasks. Each column shows the mean gaze estimation for each mode after the procedure done before the gaze pointing task, i.e. after the initial *Pupil* calibration, the *Pupil* re-calibration and taking off the device and putting it back on.

after taking the device off and putting it on again, is shown. Note that we conducted no re-calibration of the *Pupil* eye tracker in the gaze pointing task after this procedure. We noticed the lowest error for *hEYEbrid* after the initial *Pupil* calibration (M=2.16°, SD=0.72°), the re-calibration of the *Pupil* eye tracker (M=2.09°, SD=0.79°) and the simulated calibration drift (M=2.36°, SD=0.80°). We conducted independent-samples t-tests for all pause combinations for each mode. We noticed a significant difference in gaze estimation error between re-calibration (M=2.61°, SD=2.03°) and taking off/putting back the device (M=3.95°, SD=1.85°) for *Pupil* ($t(48)$=-3.27, p<0.05). We did not find further significant differences for the other combinations and modes.

5 DISCUSSION

Our results show that combining the corneal reflection within the pupil together with the pupil center in a hybrid approach achieves an average gaze estimation accuracy of 2.19°, compared to 3.09° for *Pupil* and 4.05° for *3C-hEYEbrid*. This supports our initial assumption that the pupil center, mapped from the infrared eye image on the reflected environment (i.e the corneal image), converges with the actual gaze point in the real scene.

With *hEYEbrid*, we are able to achieve a higher gaze estimation accuracy than existing approaches using corneal reflections in a mobile system. Takemura et al. [63] built a wearable device using a mobile phone. Their approach is based on corneal images and a model-based tracking approach for gaze estimation. They evaluated their system in a static single display scenario (24-inch screen), in which the participant's head was fixed at a distance of 70 cm from the screen. There, they were able to achieve 9.5° gaze estimation error on average.
Nitschke et al. [35] also built a head-mounted device using a corneal imaging camera. To realize gaze estimation they rely on 3D eye pose estimation. Similar to [63], they evaluated their approach in a single-display scenario (23-inch screen), in which participants were sitting 60 cm away from the monitor. They were able to achieve a gaze estimation error of 2.51°.
With *hEYEbrid*, we developed a method that enables accurate gaze estimation in a mobile and pervasive interaction setting. In our evaluation, we set up a realistic environment using a screen as well as tangible objects as gaze targets. Thus we simulated a more dynamic yet realistic setting. We intentionally choose this kind of accuracy evaluation, instead of doing a simple 9-point accuracy test on a screen. Nevertheless, we implemented a similar accuracy test by including the screen as one of the gaze targets in the task. Note that we performed even better than *Pupil* using marker tracking for gaze estimation on a screen, since we achieved an average gaze estimation error of 1.64°compared to 2.84°. We noticed an even larger error for *Pupil* on the tangible gaze targets of 3.27°. This error stems from the 9-point calibration itself, which is a mixture of operator and system-controlled procedure. It was done on the laptop screen only once. We did not validate the calibration to account for correction. We did so to create a realistic baseline for comparison. In doing so, we underline the drawback of this type of calibration, as already shown in [45], and highlight the advantage of *hEYEbrid*. How error-prone and uninfluenceable the calibration process is can be also seen on the results during the task after initial calibration and after the re-calibration of the *Pupil* Labs eye tracker. We noticed a better gaze estimation accuracy for the latter of the two, although the participants did the same gaze pointing task in both cases.
Furthermore, the developed method can be considered as an advance in comparison to existing approaches that realize high gaze estimation accuracy [25]. This approach relies on an active user calibration that has to be renewed at regular time intervals. *hEYEbrid* is not influenced by a calibration drift caused by taking off the head-mounted device.

We did not found a significant effect on gaze estimation for *hEYEbrid* when taking off the device and putting it back on. However, we noticed a slightly worse result. This may be caused by minimal changes in the image quality of the corneal images or less accurate pupil tracking. This also has a negative effect on the results for

3C-hEYEbrid, as it builds on the output of *hEYEbrid*. As expected, the Pupil Labs eye tracker achieves much worse results caused by the simulated calibration drift, which invalidates the mapping function (as shown in [26]).

3C-hEYEbrid uses the two eye cameras plus an additional scene camera and is able to compute the gaze in the scene by matching the corneal image onto the scene image. Note that the detection of objects via AR markers and the display tracking via natural features is performed in the scene images. It achieves less accurate results than the other methods, except for the on-screen targets. In this case an average gaze estimation error of 2.17° was achieved, compared to 2.83° for *Pupil* and 1.64° for *hEYEbrid*. A possible explanation is that *3C-hEYEbrid* relies on natural feature tracking to update its frame-based calibration. To match the pupil center into the scene image, it needs to compute the homography matrix between the corneal and scene camera images. Hence the method does two transformations (from IR to corneal and from corneal to scene images) that might introduce errors. Note that we also did not account for any corrections of the images; for instance, we did no undistortion of corneal images. The study was conducted in a minimally furnished room. Therefore the image features were very limited, except when looking at the projected screen, as it showed feature-rich content. Thus our setup shows the limitations of *3C-hEYEbrid* by design. However, when enough features are available, this method also performs well, indicated by the good results for the on-screen targets (2.84° on average). Nevertheless, this approach relies on three cameras, of which the scene camera especially is an issue, as it can cause discomfort for other people.
In our experiment we showed the applicability of our approach in a mobile indoor lab setting. Although we did not enforce constraints such as remaining seated and keeping the head still, we controlled the correct initial setup of the device (i.e. correct camera-eye distance). Consequently the results show the potential of using *hEYEbrid* in indoor settings, such as retail stores, shopping centers or airports. We also believe, that the concept is usable in outdoor settings to a certain extent. The current implementation calls for at least stable lighting conditions (e.g. sun or partially cloudy).

6 MOBILE HEYEBRID

The results of the conducted user study showed that connecting an infrared eye and a corneal camera in a hybrid approach is suitable for user-calibration-free gaze estimation. Thus, we designed and implemented a mobile and wearable eye gaze tracking system that can be used to conduct eye-tracking experiments in the wild and to build interactive systems. For this, we connected the head-mounted device, equipped with the two eye cameras, to a mobile phone. We developed a *hEYEbrid* mobile application that is able to drive the two eye cameras. Moreover, it takes care of the pupil tracking as well as the mapping of the pupil from IR to corneal images, as described earlier. Basically, one can spontaneously start to use the system in everyday life settings. The application is able to record the camera streams as well as the information about the pupil positions, which can be imported into the Pupil Labs player software for later analysis and processing.

Figure 1 illustrates the application's interface, which we designed to be easy to use. It guides the user through the setup of the system, showing the required hardware parts (the mobile phone and the head-mounted device). The main screen of the application is divided into three expandable views, integrated in a list view. The control view is used to inform the user about the connection and tracking state. Here, several plugins can be activated, like gaze estimation on screens. Two more views – the IR and Corneal views – show the camera streams and give direct feedback about the processing, i.e. the pupil tracking and mapping. This is of great importance, as it provides the user the ability to adjust the eye-camera distance to get sharp corneal images, which at the same time results in the optimal mapping. In this way we overcome one of the aformentioned limitations. The application can be easily extended by several functionalities which make use of the computed gaze point and the corneal reflection.
Besides the *hEYEbrid* mode, we also implemented mobile versions for *3C-hEYEbrid* and Pupil Labs monocular

eye trackers. With a fully equipped head-mounted device, i.e. three cameras (infrared, corneal and scene) one can decide which mode to use. However, keep in mind that only *hEYEbrid* and *3C-hEYEbrid* are calibration free in that they only need to establish a camera mapping once unlike the Pupil Labs eye tracker.

6.1 Applications

Mobile calibration-free gaze estimation opens up the space for a variety of application cases. We distinguish between the usage of *hEYEbrid* as an analysis tool and an input device.
Coming back to the introduction, there is increasing interest in the integration of gaze tracking in AR and VR devices. Both the reflection of an AR glasses' screen (e.g. HoloLens) as well as that of a VR headset's display are visible on the corneal images. The availability of information about where a person is looking in tasks such as industrial maintenance will be of great benefit to guide a worker, for example. A virtual reality headset offers the ideal conditions to integrate *hEYEbrid*, since nothing other than the screen content (i.e. the VR environment) is reflected on the human cornea. Moreover, the distance between the VR headset's display and the eye is fixed, thus we do not have to deal with blurry corneal images due to defocus. However, the current camera clip would have to be scaled down to fit into a headset. Possible use cases range from foveated rendering [47] to hands-free interaction.
hEYEbrid also enables the creation of interactive applications at pervasive scale. A person's gaze, which is available all times, can be shared with connected devices in the environment. Gaze estimation on public displays can be used to enable the development of interactive screens. A person's gaze can be used as a direct or indirect input modality to control the screen by pointing, or support the user in reading text on a public display, like in [28].

Besides utilizing *hEYEbrid* for creating gaze-based interfaces, corneal images can be analyzed to gain insights in human cognitive processes. That is, *hEYEbrid* enables gaze estimation for real-life in-the-wild scenarios. The mobile system can be used to conduct user studies in a lightweight manner. Multiple persons can be equipped with the system, to collect a large amount of data in parallel. This data can be analyzed post-hoc for different application use cases, such as landmark extraction [29] or activity analysis [18, 57]. Since our daily behavior is reflected on our eyes, corneal images can serve as a basis for camera-based lifelogging [27]. If a person is fixating an object, this behavior is reflected in the corneal image. For example, such information can be used for attention measurements.

To demonstrate the potentials of *hEYEbrid*, we integrated two example plugins: gaze estimation on public displays, and automatic object detection, reflected on the user's eye within the area of the pupil. Both example applications are shown in the video figure.

6.2 Implementation

The architecture of the final mobile prototype is shown in Figure 13. It consists of two main components on the hardware side: (1) the head-mounted device and (2) a Nexus 6P phone, running Android 7.1, to drive the device. All cameras are attached to a USB 2.0 hub that is directly connected to the phone via USB-C. There is no need for an extra power source to power two cameras. However, to run *3C-hEYEbrid* or a Pupil Labs eye tracker, an additional power source has to be used. Running the application on the phone, its battery lasts for approximately five hours. Note that during this measurement, Bluetooth, WIFI and GPS were also enabled. We will explain the software architecture only for *hEYEbrid* in the following. The other modes are developed in the same manner.
The software architecture of the Android application consists of three main parts: (1) the backend to process all images, (2) the frontend for live visualization and to control the app, and (3) the plugins to add advanced functionalities. The *hEYEbrid* application mode is developed as a native Android application, based on the Android

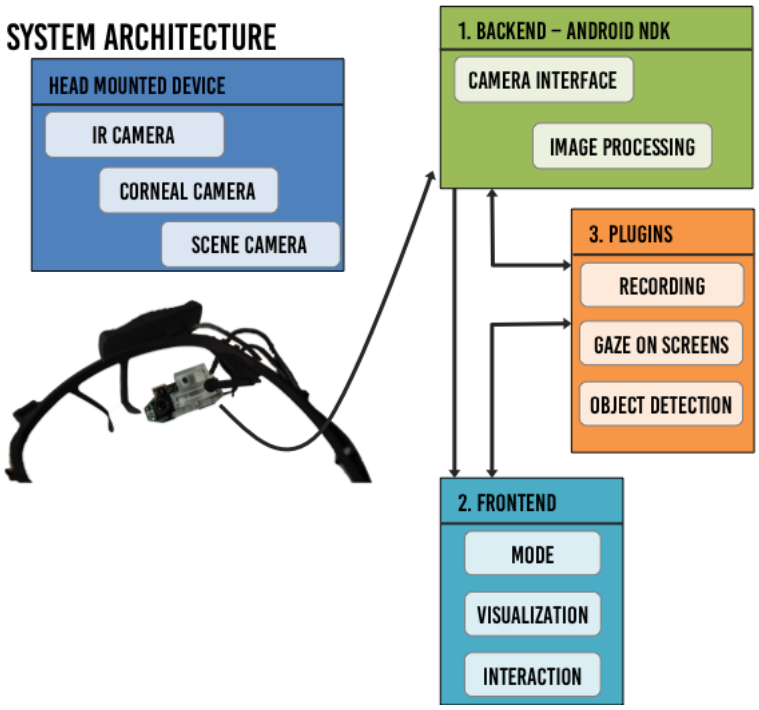

Fig. 13. Architecture of the mobile system usable with *hEYEbrid*, *3C-hEYEbrid* and a Pupil Labs eye tracker. The hardware is made up of a head-mounted device containing two or three cameras, depending on the mode. The software side contains three main parts.

SDK 25. The backend parts are developed in C++ and included via Android's native development kit (NDK), version 7. To stream both camera streams at once over USB, we use libuvc, a cross-platform library for USB video devices. In particular we include an existing library, wrapping libuvc[13], for usage with Android. This library allows us to set the available camera settings (e.g., frame rate, resolution, brightness) and to control the bandwidth factor. The IR camera frames are streamed with a resolution of 640 x 480 pixels, corneal images with a resolution of 1280 x 960 px, both at 30Hz. For image processing we use the Java native interface to realize the pupil tracking. We included PUPIL's built-in pupil detector (defined in detect_2d.hpp) and all its dependencies. For this, we had to include specific versions of Eigen[14] and TBB[15] for Android. The method returns an ellipse describing the pupil structure. This is transformed onto the corneal image applying the homography matrix that was computed beforehand during construction. For the projection we make use of the OpenCV 3.2 library for C++. Finally, the backend offers a callback delivering the pupil center and the cut-out corneal image.
The frontend implements three views in different panels. We chose a list-like visualization of the views and included the ExpandableLayout library[16] to smoothly expand and collapse the views. We used Android's Material design for all UI components.

[13]https://github.com/saki4510t/UVCCamera
[14]http://eigen.tuxfamily.org
[15]https://www.threadingbuildingblocks.org/tbb-mobile
[16]https://github.com/cachapa/ExpandableLayout

To integrate advanced functionalities, the application provides interfaces to the backend and frontend usable by plugins. We implemented two example plugins: For gaze estimation on displays, we use a similar approach to GazeProjector [25]. The idea is to compute the spatial relationship between the head-mounted device and the surrounding displays. In contrast to existing approaches [70], we use the cropped corneal image as a template, to look for on the display's content. In both images key features are extracted using AKAZE [1] and matched using a 2-nn Brute-Force matcher. To estimate the spatial relationship, the found key feature pairs are used to compute a homography matrix. For the algorithm, the screen's content is streamed via screenshots from the display to the mobile application. For this a script has to be executed on the computer connected to the display. It resizes the screenshots to 480 x 360 pixels and pushes them onto the mobile phone. The mobile application sends back the estimated gaze position to use this for further processing.
For object extraction, we use the YOLO[17] framework for real-time object detection [48] based on neural networks. We use the pre-trained weight file (yolo.weights) that is already available within the framework and the standard configuration (yolo.cfg). To speed up the processing, the cropped corneal images are sent as batches to a server that does the processing. After a successful object detection, the result with the highest detection probability, closest to the pupil center, is sent back to the mobile phone.
The main software part (i.e., the backend) runs at 25-30 fps on the Android phone (Nexus 6P). The gaze estimation plugin has to do some more complex computations (e.g., feature matching) and thus runs at 25 fps. The object detection plugin is only limited by the transfer rate of the images to the processing server, as YOLO is able to process the stream at up to 90 fps. Hence the plugin runs at at 25-30 fps as the main program.
We are going to make the Android application publicly available through GitHub including a guide and necessary files to re-build the hardware.

7 LIMITATIONS

Apart from its numerous advantages over state-of-the-art eye tracking systems, *hEYEbrid* also comes with some limitations. Our concept of gaze estimation requires two cameras, bundled into one enclosure that is mounted in front of the user's eye. The construction may interfere with the field of view. However, using a dedicated hardware prototype with both cameras on one board and smaller cameras could reduce its total size. This would make the head-mounted device even smaller, lighter in weight and more comfortable.
Further, we used AR markers for object detection in our experiment, to robustly track the objects. Nevertheless, we believe that with a higher resolution eye camera, objects are able to be detected via feature tracking or a trained classifier.
Currently, computing the mapping between the infrared and corneal camera image plane is the biggest limitation. As described above, we rely on a homography matrix that computes a planar mapping in 2D. If the distance between the camera and the user's eye changes to a great extent, the mapping will become more incorrect. However, we handle this limitation in the following way. To compute the homography matrix we record images from both cameras at a fixed distance with sharp focus. Consequently, we obtain an optimal mapping in this setting. When putting on the device, the camera is placed in such a way, that the focus is also sharp. Hence, the distance between cameras and eye is nearly the same as during the mapping step. The good results of our experiment after taking off and putting back on the device support this assumption. In a totally uncontrolled environment the user has to make sure to place the camera at a focused distance from the eye. It would be possible to support that procedure by computing the current focus value of the corneal image (e.g. by using depth sensors). In this way it might be possible to guide the user in placing the camera at the right distance from the eye.
Besides camera-eye distance, eye-object distance is the second source of blurry corneal images. That is, if the user is focusing at an object at a far distance, its representation on the corneal image is rather unsharp, caused by

[17] https://pjreddie.com/darknet/yolo/

the mirror properties. Consequently our current implementation has a certain working distance. Using cameras with auto-focus may help to circumvent this limitation. Another possibility is to use neural networks that are specifically trained on such blurry images to handle these cases.
We have little knowledge about the long-term accuracy of *hEYEbrid*. The experiment took thirty minutes at maximum per participant, caused by longer pauses between the tasks. Consequently the presented results are representative for a usage time within half an hour. It is noteworthy that the homography matrix between both cameras was not updated during the complete experiment, which lasted over several days.
Also, *hEYEbrid* has difficulties working in dark environments, as nearly no information will be reflected on the cornea. Installing cameras with a higher ISO sensitivity could help to make information visible on the corneal images, even in darker settings. Also, changes in lighting might affect the video quality of corneal images and thus the processing. Since the human pupil is sensitive to light, a bright environment will have a miotic effect (i.e. shrinking of the pupil). This directly influences the size and resolution of the extracted corneal image by *hEYEbrid*. However, the approach might still work, as we rely only on the pupil center that gives us the gaze point in the reflected environment. The extracted field of view will be smaller, but sufficient to map the gaze on smaller objects (e.g. a mobile phone), whereas it will be insufficient for large objects (e.g. cars, buildings).
In summary it makes sense to explore cameras with different properties and configurations, especially to make it robust for outdoor usage. Note that when integrating a camera that offers capabilities like auto-focus, auto-iris and high ISO sensitivity, the complete device will get more expensive. Nevertheless, we believe that *hEYEbrid* is a promising concept for user-calibration-free gaze estimation in pervasive settings to enable spontaneous gaze-based interaction as well as eye-tracking experiments in the wild.

8 CONCLUSION & FUTURE WORK

In this paper, we presented *hEYEbrid*, an approach for calibration-free and very accurate gaze estimation in pervasive settings. In contrast to existing systems, *hEYEbrid* combines the images from an infrared eye and a corneal camera in a hybrid approach and works without any prior user calibration. Hence, its accuracy is not influenced by calibration drifts. The concept is integrated into Pupil Lab's open-source eye-tracking framework, by adding a corneal camera to the head-mounted device and developing plugins for the capture and player software.

We conducted a user study in which we used a mobile setup to evaluate *hEYEbrid*'s gaze estimation accuracy against a state-of-the-art Pupil Labs eye tracker and *3C-hEYEbrid*, an extended version using the scene camera as additional image information. We found that our approach performs better than the well-established Pupil Labs eye tracker. In addition, it compares well to existing approaches based on corneal imaging. The results are very promising and underline the potential to realize mobile gaze-based interaction in everyday life settings.

As a first step towards this vision, we built a mobile and wearable system which is able to react in real time. This system, which consists of a head-mounted device (with the two eye cameras) and an Android phone, is usable in different application settings. Researchers can profit from this device, as they can use it for eye-tracking experiments in unconstrained environments. Besides, the device can be used to build gaze-based interfaces in pervasive settings. We developed two example applications to showcase the potential of *hEYEbrid*. But, as discussed in the limitation section above, there is lot of room for future work to resolve current issues and improve the approach. The next step is to evaluate the system in an in-the-wild study. We plan to equip multiple persons with the device and ask them to use it during the day while doing several activities indoors and outdoors. Thereby we want to investigate the long-term usage on the one side and the influence of environmental factors (e.g., lighting). In addition we would like to test prototypes with cameras having different properties to resolve the issue of defocus and unsharp images. Finally, having a system that can be used in fully unconstrained settings

for long-term data recording, will bring further insights into people's gaze behavior, and create novel interactions that could benefit from such a device.

9 ACKNOWLEDGEMENTS

Special thanks go to Andreas Bulling, who was involved in the initial discussions of the approach.

REFERENCES

[1] Pablo Fernández Alcantarilla, Jesús Nuevo, and Adrien Bartoli. 2013. Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces. In *BMVC*.

[2] Christian Nitschke Atsushi Nakazawa and Toyoaki Nishida. 2016. Registration of Eye Reflection and Scene Images using an Aspherical Eye Model. *Journal of the Optical Society of America A* 33, 11 (2016), 2264. https://doi.org/10.1364/JOSAA.33.002264

[3] Andreas Bulling and Hans Gellersen. 2010. Toward Mobile Eye-Based Human-Computer Interaction. *IEEE Pervasive Computing* 9, 4 (Oct. 2010), 8–12. https://doi.org/10.1109/MPRV.2010.86

[4] Juan J. Cerrolaza, Arantxa Villanueva, and Rafael Cabeza. 2012. Study of Polynomial Mapping Functions in Video-Oculography Eye Trackers. *ACM Trans. Comput.-Hum. Interact.* 19, 2, Article 10 (July 2012), 25 pages. https://doi.org/10.1145/2240156.2240158

[5] Juan J. Cerrolaza, Arantxa Villanueva, Maria Villanueva, and Rafael Cabeza. 2012. Error Characterization and Compensation in Eye Tracking Systems. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12)*. ACM, New York, NY, USA, 205–208. https://doi.org/10.1145/2168556.2168595

[6] Jixu Chen and Qiang Ji. 2011. Probabilistic Gaze Estimation Without Active Personal Calibration. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '11)*. IEEE Computer Society, Washington, DC, USA, 609–616. https://doi.org/10.1109/CVPR.2011.5995675

[7] American Heritage Dictionary. 2007. *The American Heritage Medical Dictionary*. Houghton Mifflin Company.

[8] Andrew T. Duchowski. 2007. *Eye Tracking Methodology: Theory and Practice*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[9] Wolfgang Fuhl, Thomas Kübler, Katrin Sippel, Wolfgang Rosenstiel, and Enkelejda Kasneci. 2015. *ExCuSe: Robust Pupil Detection in Real-World Scenarios*. Springer International Publishing, Cham, 39–51. https://doi.org/10.1007/978-3-319-23192-1_4

[10] 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

[11] Wolfgang Fuhl, Marc Tonsen, Andreas Bulling, and Enkelejda Kasneci. 2016. Pupil Detection for Head-Mounted Eye Tracking in the Wild:An Evaluation of the State of the Art. *Machine Vision and Applications* 27, 8 (01 Nov 2016), 1275–1288. https://doi.org/10.1007/s00138-016-0776-4

[12] Lotfi El Hafi, Tsukasa Ogasawara, Ming Ding, and Jun Takamatsu. 2016. Gaze Tracking Using Corneal Images Captured by a Single High-Sensitivity Camera. (01 2016), 33–43. https://doi.org/10.1049/ibc.2016.0033

[13] Riad Hammoud. 2008. *Passive Eye Monitoring: Algorithms, Applications and Experiments* (1st ed.). Springer Publishing Company, Incorporated.

[14] Dan Witzner Hansen and Qiang Ji. 2010. In the Eye of the Beholder: A Survey of Models for Eyes and Gaze. *IEEE Trans. Pattern Anal. Mach. Intell.* 32, 3 (March 2010), 478–500. https://doi.org/10.1109/TPAMI.2009.30

[15] R. I. Hartley and A. Zisserman. 2004. *Multiple View Geometry in Computer Vision* (2nd ed.). Cambridge University Press, ISBN: 0521540518.

[16] K. Holmqvist, M. Nyström, R. Andersson, R. Dewhurst, H. Jarodzka, and J. van de Weijer. 2011. *Eye Tracking: A Comprehensive Guide to Methods and Measures*. Oxford: Oxford University Press.

[17] Michael Xuelin Huang, Tiffany C.K. Kwok, Grace Ngai, Stephen C.F. Chan, and Hong Va Leong. 2016. Building a Personalized, Auto-Calibrating Eye Tracker from User Interactions. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 5169–5179. https://doi.org/10.1145/2858036.2858404

[18] Yoshio Ishiguro, Adiyan Mujibiya, Takashi Miyaki, and Jun Rekimoto. 2010. Aided Eyes: Eye Activity Sensing for Daily Life. In *Proceedings of the 1st Augmented Human International Conference (AH '10)*. ACM, New York, NY, USA, Article 25, 7 pages. https://doi.org/10.1145/1785455.1785480

[19] Robert J. K. Jacob. 1990. What You Look at is What You Get: Eye Movement-Based Interaction Techniques. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '90)*. ACM, New York, NY, USA, 11–18. https://doi.org/10.1145/97243.97246

[20] Amir-Homayoun Javadi, Zahra Hakimi, Morteza Barati, Vincent Walsh, and Lili Tcheang. 2015. SET: A Pupil Detection Method using Snusoidal Approximation. *Frontiers in Neuroengineering* 8 (2015), 4. https://doi.org/10.3389/fneng.2015.00004

[21] Qiang Ji and Xiaojie Yang. 2002. Real-Time Eye, Gaze, and Face Pose Tracking for Monitoring Driver Vigilance. *Real-Time Imaging* 8, 5 (2002), 357–377. https://doi.org/10.1006/rtim.2002.0279

[22] Moritz Kassner, William Patera, and Andreas Bulling. 2014. Pupil: An Open Source Platform for Pervasive Eye Tracking and Mobile Gaze-Based Interaction. In *Adjunct Proceedings of UbiComp 2014 (UbiComp '14 Adjunct)*. ACM, New York, NY, USA, 1151–1160. https://doi.org/10.1145/2638728.2641695

[23] Arie E. Kaufman, Amit Bandopadhay, and Bernard D. Shavi. 1993. An Eye Tracking Computer User Interface.

[24] Pupil Labs. 2017. Pupil Labs Store. (2017). Retrieved Oct 2017 from https://pupil-labs.com/store/

[25] Christian Lander, Sven Gehring, Antonio Krüger, Sebastian Boring, and Andreas Bulling. 2015. GazeProjector: Accurate Gaze Estimation and Seamless Gaze Interaction Across Multiple Displays. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. ACM, New York, NY, USA, 395–404. https://doi.org/10.1145/2807442.2807479

[26] Christian Lander, Frederic Kerber, Thorsten Rauber, and Antonio Krüger. 2016. A Time-Efficient Re-Calibration Algorithm for Improved Long-Term Accuracy of Head-Worn Eye Trackers. In *Proceedings of the Ninth ACM Symposium on Eye Tracking Research & Applications (ETRA '16)*. ACM, New York, NY, USA, 213–216. https://doi.org/10.1145/2857491.2857513

[27] Christian Lander, Antonio Krüger, and Markus Löchtefeld. 2016. "The Story of Life is Quicker Than the Blink of an Eye": Using Corneal Imaging for Life Logging. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct (UbiComp '16)*. ACM, New York, NY, USA, 1686–1695. https://doi.org/10.1145/2968219.2968337

[28] Christian Lander, Marco Speicher, Denise Paradowski, Norine Coenen, Sebastian Biewer, and Antonio Krüger. 2015. Collaborative Newspaper: Exploring an Adaptive Scrolling Algorithm in a Multi-user Reading Scenario. In *Proceedings of the 4th International Symposium on Pervasive Displays (PerDis '15)*. ACM, New York, NY, USA, 163–169. https://doi.org/10.1145/2757710.2757734

[29] Christian Lander, Frederik Wiehr, Nico Herbig, Antonio Krüger, and Markus Löchtefeld. 2017. Inferring Landmarks for Pedestrian Navigation from Mobile Eye-Tracking Data and Google Street View. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '17)*. ACM, New York, NY, USA, 2721–2729. https://doi.org/10.1145/3027063.3053201

[30] Dongheng Li, David Winfield, and Derrick J Parkhurst. 2005. Starburst: A Hybrid Algorithm for Video-Based Eye Tracking Combining Feature-Based and Model-Based Approaches. In *Computer Vision and Pattern Recognition-Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on*. IEEE, 79–79. https://doi.org/10.1109/CVPR.2005.531

[31] David G. Lowe. 2004. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vision* 60, 2 (Nov. 2004), 91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94

[32] Carlos H. Morimoto and Marcio R. M. Mimica. 2005. Eye Gaze Tracking Techniques for Interactive Applications. *Comput. Vis. Image Underst.* 98, 1 (April 2005), 4–24. https://doi.org/10.1016/j.cviu.2004.07.010

[33] Samuel Arba Mosquera, Shwetabh Verma, and Colm McAlinden. 2015. Centration Axis in Refractive Surgery. *Eye and Vision* 2, 1 (24 Feb 2015), 4. https://doi.org/10.1186/s40662-015-0014-6

[34] Atsushi Nakazawa and Christian Nitschke. 2012. Point of Gaze Estimation Through Corneal Surface Reflection in an Active Illumination Environment. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part II (ECCV'12)*. Springer-Verlag, Berlin, Heidelberg, 159–172. https://doi.org/10.1007/978-3-642-33709-3_12

[35] Atsushi Nakazawa, Christian Nitschke, and Toyoaki Nishida. 2015. Non-Calibrated and Real-Time Human View Estimation Using a Mobile Corneal Imaging Camera. In *Multimedia & Expo Workshops (ICMEW), 2015 IEEE International Conference on*. IEEE, 1–6. https://doi.org/10.1109/ICMEW.2015.7169846

[36] Karlene Nguyen, Cindy Wagner, David Koons, and Myron Flickner. 2002. Differences in the Infrared Bright Pupil Response of Human Eyes. In *Proceedings of the 2002 Symposium on Eye Tracking Research & Applications (ETRA '02)*. ACM, New York, NY, USA, 133–138. https://doi.org/10.1145/507072.507099

[37] K. Nishino, P. N. Belhumeur, and S. K. Nayar. 2005. Using Eye Reflections for Face Recognition under Varying Illumination. In *IEEE ICCV 2005*, Vol. 1. IEEE, 519–526. https://doi.org/10.1109/ICCV.2005.243

[38] K. Nishino and S. Nayar. 2004. Eyes for Relighting. In *Proc. SIGGRAPH 2004 (SIGGRAPH '04)*. ACM, New York, NY, USA, 704–711. https://doi.org/10.1145/1186562.1015783

[39] K. Nishino and S. Nayar. 2004. The World in an Eye. In *In IEEE CVPR, Volume I*. 444–451. https://doi.org/10.1109/CVPR.2004.1315066

[40] K. Nishino and S. Nayar. 2006. Corneal Imaging System: Environment from Eyes. *International Journal of Computer Vision* 70, 1 (2006), 23–40. https://doi.org/10.1007/s11263-006-6274-9

[41] Christian Nitschke, Atsushi Nakazawa, and Toyoaki Nishida. 2013. I See What You See: Point of Gaze Estimation from Corneal Images. In *Proceedings of the 2013 2nd IAPR Asian Conference on Pattern Recognition (ACPR '13)*. IEEE Computer Society, Washington, DC, USA, 298–304. https://doi.org/10.1109/ACPR.2013.84

[42] C. Nitschke, A. Nakazawa, and H. Takemura. 2011. Display-Camera Calibration using Eye Reflections and Geometry Constraints. *Computer Vision and Image Understanding* 115, 6 (2011), 835–853. https://doi.org/10.1016/j.cviu.2011.02.008

[43] Christian Nitschke, Atsushi Nakazawa, and Haruo Takemura. 2011. Image-Based Eye Pose and Reflection Analysis for Advanced Interaction Techniques and Scene Understanding. *Computer Vision and Image Media (CVIM)* (2011), 1–16. http://ci.nii.ac.jp/naid/110008584047/en/

[44] Christian Nitschke, Atsushi Nakazawa, and Haruo Takemura. 2013. Corneal Imaging Revisited: An Overview of Corneal Reflection Analysis and Applications. *Information and Media Technologies* 8, 2 (2013), 389–406. https://doi.org/10.11185/imt.8.389

[45] Marcus Nyström, Richard Andersson, Kenneth Holmqvist, and Joost van de Weijer. 2013. The Influence of Calibration Method and Eye Physiology on Eyetracking Data Quality. *Behavior Research Methods* 45, 1 (01 Mar 2013), 272–288. https://doi.org/10.3758/s13428-012-0247-4

[46] 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

[47] Daniel Pohl, Xucong Zhang, Andreas Bulling, and Oliver Grau. 2016. Concept for Using Eye Tracking in a Head-mounted Display to Adapt Rendering to the User's Current Visual Field. In *Proceedings of the 22Nd ACM Conference on Virtual Reality Software and Technology (VRST '16)*. ACM, New York, NY, USA, 323–324. https://doi.org/10.1145/2993369.2996300

[48] Joseph Redmon and Ali Farhadi. 2016. YOLO9000: Better, Faster, Stronger. (2016). arXiv:arXiv:1612.08242

[49] David A. Robinson. 1963. A Method of Measuring Eye Movement Using a Scleral Search Coil in a Magnetic Field. *IEEE Transactions on Bio-medical Electronics* 10 (1963), 137–45. https://doi.org/10.1109/TBMEL.1963.4322822

[50] 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

[51] Dirk Schnieders, Xingdou Fu, and Kwan-Yee K Wong. 2010. Reconstruction of Display and Eyes from a Single Image. In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1442–1449. https://doi.org/10.1109/CVPR.2010.5539799

[52] Susan K. Schnipke and Marc W. Todd. 2000. Trials and Tribulations of Using an Eye-tracking System. In *CHI '00 Extended Abstracts on Human Factors in Computing Systems (CHI EA '00)*. ACM, New York, NY, USA, 273–274. https://doi.org/10.1145/633292.633452

[53] D. Scott, J.M. Findlay, Winchester Hursley Human Factors Laboratory, and IBM UK Hursley Human Factors Laboratory. 1991. *Visual Search, Eye Movements and Display Units*. IBM UK Hursley Human Factors Laboratory.

[54] SMI 2017. *SMI Eye Tracking Glasses 2*. SMI. https://www.smivision.com/wp-content/uploads/2017/05/smi_prod_ETG_120Hz_asgm.pdf.

[55] Dave M. Stampe. 1993. Heuristic Filtering and Reliable Calibration Methods for Video-Based Pupil-Tracking Systems. *Behavior Research Methods, Instruments, & Computers* 25, 2 (01 Jun 1993), 137–142. https://doi.org/10.3758/BF03204486

[56] Julian Steil and Andreas Bulling. 2015. Discovery of Everyday Human Activities from Long-term Visual Behaviour Using Topic Models. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. ACM, New York, NY, USA, 75–85. https://doi.org/10.1145/2750858.2807520

[57] J. Steil and A. Bulling. 2015. Discovery of Everyday Human Activities From Long-term Visual Behaviour Using Topic Models. In *Proc. UbiComp 2015*. 75–85. https://doi.org/10.1145/2750858.2807520

[58] Yusuke Sugano and Andreas Bulling. 2015. Self-Calibrating Head-Mounted Eye Trackers Using Egocentric Visual Saliency. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software; Technology (UIST '15)*. ACM, New York, NY, USA, 363–372. https://doi.org/10.1145/2807442.2807445

[59] Yusuke Sugano, Yasuyuki Matsushita, and Yoichi Sato. 2013. Appearance-Based Gaze Estimation Using Visual Saliency. *IEEE Trans. Pattern Anal. Mach. Intell.* 35, 2 (Feb. 2013), 329–341. https://doi.org/10.1109/TPAMI.2012.101

[60] 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

[61] Lech Świrski and Neil Dodgson. 2013. A Fully-Automatic, Temporal Approach to Single Camera, Glint-Free 3D Eye Model Fitting [Abstract]. In *Proceedings of ECEM 2013*. http://www.cl.cam.ac.uk/research/rainbow/projects/eyemodelfit/

[62] Kentaro Takemura, Shunki Kimura, and Sara Suda. 2014. Estimating Point-of-Regard Using Corneal Surface Image. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '14)*. ACM, New York, NY, USA, 251–254. https://doi.org/10.1145/2578153.2578197

[63] Kentaro Takemura, Tomohisa Yamakawa, Jun Takamatsu, and Tsukasa Ogasawara. 2013. Estimating Focused Object Using Corneal Surface Image for Eye-Based Interaction. In *3rd International Workshop on Pervasive Eye Tracking and Mobile Eye-Based Interaction*.

[64] Techcrunch. 2016. Google Buys Eyefluence Eye-Tracking Startup. (2016). Retrieved Oct 2017 from https://techcrunch.com/2016/10/24/google-buys-eyefluence-eye-tracking-startup/

[65] Techcrunch. 2017. Apple Acquires SMI Eye-Tracking Company. (2017). Retrieved Oct 2017 from https://techcrunch.com/2017/06/26/apple-acquires-smi-eye-tracking-company/

[66] Tobii Technology. 2017. Tobii Pro Glasses 2. (2017). Retrieved Oct 2017 from https://www.tobiipro.com/product-listing/tobii-pro-glasses-2/

[67] Tobii 2015. *Pro Glasses 2 Manual*. Tobii. https://www.tobiipro.com/siteassets/tobii-pro/user-manuals/tobii-pro-glasses-2-user-manual.pdf/?v=1.20.2.

[68] Tobii Gaming 2017. *Tobii 4C Calibration*. Tobii Gaming. https://help.tobii.com/hc/en-us/articles/209530409-Create-a-new-user-profile.

[69] Tobii Gaming 2017. *Tobii 4C Controller*. Tobii Gaming. https://tobiigaming.com/eye-tracker-4c.

[70] Jayson Turner, Jason Alexander, Andreas Bulling, and Hans Gellersen. 2015. Gaze+RST: Integrating Gaze and Multitouch for Remote Rotate-Scale-Translate Tasks. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 4179–4188. https://doi.org/10.1145/2702123.2702355

[71] Roel Vertegaal et al. 2003. Attentive User Interfaces. *Commun. ACM* 46, 3 (2003), 30–33.

[72] A. Villanueva and R. Cabeza. 2008. A Novel Gaze Estimation System With One Calibration Point. *Trans. Sys. Man Cyber. Part B* 38, 4 (Aug. 2008), 1123–1138. https://doi.org/10.1109/TSMCB.2008.926606

[73] Laurence R. Young and David Sheena. 1975. Survey of Eye Movement Recording Methods. *Behavior Research Methods & Instrumentation* 7, 5 (01 Sep 1975), 397–429. https://doi.org/10.3758/BF03201553

[74] Lawrence H Yu and Moshe Eizenman. 2004. A New Methodology for Determining Point-of-Gaze in Head-Mounted Eye Tracking Systems. *IEEE transactions on bio-medical engineering* 51, 10 (October 2004), 1765âĂŤ1773. https://doi.org/10.1109/tbme.2004.831523

[75] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. 2015. Appearance-Based Gaze Estimation in the Wild. *CoRR* abs/1504.02863 (2015). http://arxiv.org/abs/1504.02863

Received August 2017; accepted October 2017