CrowdEyes: Crowdsourcing for Robust Real-World Mobile Eye Tracking

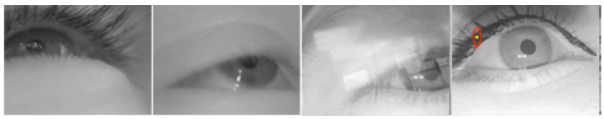
Mohammad Othman[1], Telmo Amaral[1], Róisín McNaney[2], Jan D. Smeddinck[3], John Vines[4], Patrick Olivier[1]
[1]Open Lab, Newcastle University, Newcastle upon Tyne, UK
[2]School of Computing and Communications, Lancaster University, Lancaster, UK
[3]Digital Media Lab, TZI, University of Bremen, Bremen, Germany
[4]Northumbria University, Newcastle upon Tyne, UK
{m.othman1,telmo.amaral,Patrick.olivier}@newcastle.ac.uk, r.mcnaney@lancaster.ac.uk, smeddinck@tzi.de, john.vines@northumbria.ac.uk

ABSTRACT
Current eye tracking technologies have a number of drawbacks when it comes to practical use in real-world settings. Common challenges, such as high levels of daylight, eyewear (e.g. spectacles or contact lenses) and eye make-up, give rise to noise that undermines their utility as a standard component for mobile computing, design, and evaluation. To work around these challenges, we introduce *CrowdEyes*, a mobile eye tracking solution that utilizes crowdsourcing for increased tracking accuracy and robustness. We present a pupil detection task design for crowd workers together with a study that demonstrates the high-level accuracy of crowdsourced pupil detection in comparison to state-of-the-art pupil detection algorithms. We further demonstrate the utility of our crowdsourced analysis pipeline in a fixation tagging task. In this paper, we validate the accuracy and robustness of harnessing the crowd as both an alternative and complement to automated pupil detection algorithms, and explore the associated costs and quality of our crowdsourcing approach.

Author Keywords
Crowdsourcing; crowd quality control; eye tracking; mobile computing; wearable computing; pupil detection.

ACM Classification Keywords
H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous;

INTRODUCTION
Eye tracking is a method of measuring an individual's eye movement to identify both where a person is looking (*gaze*) and the sequence in which the person's eyes are shifting from one location to another. Eye tracking tells us about points of interest in which the person's eyes are relatively stable (*fixation*) for a minimum duration of 100-200 ms [10], as well as the rapid eye movements (*saccade*) from one fixation to another. While eye tracking techniques are diverse—from video-oculography VOG, video-based infrared IR to electrooculography EOG (for a detailed review see [18])—this paper focuses on video-based eye tracking.



(a) (b) (c) (d)

Figure 1. Challenging pupil images in real-world scenarios: (a) natural light reflection, (b) droopy eyelids, (c) spectacles and (d) eye make-up. The red circle in (d) indicates false pupil detection when wearing eye make-up.

Video-based eye tracking relies on the detection of pupil positions to estimate gaze positions from images typically delivered by off-the-shelf video cameras. The technology has been used in a multitude of clinical, research and commercial applications; from monitoring drivers' eyes to warn them of drowsiness and distraction [29,42], to skill assessment (e.g. assessing drivers and cyclists hazard perception skills [17,19]); and for clinical diagnosis (e.g. in Parkinson's [20] and autism [13]), as well as wayfinding research (e.g. to evaluate and improve guidance systems in public infrastructures [27] and indoor environments [23]). In the field of HCI, these technologies have also been used to evaluate technologies, such as the usability and safety standards for using smartphones [22] and situated displays in public spaces [3].

Despite their diverse forms and considerable potential for applications, most eye tracking studies are conducted in artificial or semi-artificial environments—either in rooms with controlled lighting (e.g. laboratories) or in virtual reality environments. While video-based eye tracking performs comparatively well in such controlled environments, the technology fails dramatically under real-world conditions [1,2,4,30]. Failure in real-world settings is mostly attributed to low pupil detection rates due to a number of factors, including: i) uncontrolled lighting [4]; ii)

MobileHCI '17, September 04-07, 2017, Vienna, Austria

ACM ISBN 978-1-4503-5075-4/17/09.
http://dx.doi.org/10.1145/3098279.3098559

pupil occlusion by the eyelid and eyelashes [4]; iii) eyewear [7] (e.g. spectacles or contact lenses); iv) eye make-up [7]; and v) motion-blur [16] (e.g. from fast eye movements during saccades). In particular, uncontrolled lighting conditions, e.g. while walking or driving, cause reflections and differences in contrast that limit the effectiveness of automated pupil detection algorithms. Since most video-based eye tracking works in the infrared light spectrum, in many mixed lighting or outdoor conditions where infrared light (e.g. sunlight) floods the eye camera(s) (Figure 1a) the automatic detection of pupil features becomes difficult. Moreover, conditions like Ptosis (pathologic eyelid drooping) cause the eyelid to partially block the pupil of the eye camera (Figure 1b), making it difficult to detect pupil features. In a similar manner, spectacles and eye make-up result in substantial and varied forms of reflections and generally high amounts of noise (see Figure 1c and 1d).

In this paper, we propose a new approach to achieve robust and accurate eye tracking measures by harnessing the crowd. Whereas the challenges facing automated pupil detection methods often restrict eye tracking to controlled environments, our proposed approach, *CrowdEyes*, offers mobile unobtrusive eye tracking with all standard metrics independent from most common pupil detection challenges. *CrowdEyes* includes a process to localize the pupil position without automated pupil detection algorithms. It begins by decomposing the collected video of the eye into single frames, marking key frames, and crowdsourcing the localization of the pupil position in these frames. The resulting positions are then used to generate standard eye tracking metrics (e.g. gaze and fixation positions and durations, saccades) using the according methods from the open-source eye tracking platform *Pupil* [12]. Fixations can then be semantically labeled by crowd workers. *CrowdEyes* is envisaged as a runtime tool operating on mobile devices alongside eye tracking hardware; however, to accommodate for the present technical limitations of mobile devices, our initial proof-of-concept data is processed offline after collection.

Our contribution is twofold. For the crowdsourcing research community, we: i) investigate and evaluate the design of crowdsourcing tasks and strategies to affordably improve mobile wearable eye tracking technologies; ii) propose a crowd task quality assurance method that enables workers to evaluate and refine their own entries; and iii) provide experimental evidence that demonstrates that such quality methods also motivate workers to improve their accuracy. Second, we contribute to mobile wearable eye tracking by: i) working around pupil detection challenges in real-world scenarios; ii) reliably localizing pupil positions; and iii) providing a tool for the mobile eye tracking research community to generate training datasets for pupil detection algorithms on demand by harnessing the crowd.

BACKGROUND

The work presented in this paper relates to several areas of research, including computer vision based eye tracking, self-reporting eye tracking, and crowdsourcing as an alternative and complement to automated detection and recognition systems.

Computer vision based eye tracking

In recent years, there has been a growing amount of research on the design and development of mobile eye tracking technologies. Studies have investigated novel pupil detection algorithms (e.g. [5,34,35,37]) and new calibration techniques (e.g. [15,33,38]), seeking robust commercial and open-source eye trackers for real-world settings. However, since most of the state-of-the-art pupil detection algorithms are based on the edge filtering approach [7], they are very susceptible to failure under the aforementioned conditions. Tonsen et al. [36] evaluated the pupil detection success rate of five state-of-the-art pupil detection algorithms: *Swirski* [34], *ExCuSe* [5], *Isophete* [37], *Gradient* [35] and *Pupil-Labs* [12] using their large and challenging real-world *Labeled Pupil in the Wild* (LPW) dataset [36] of 130,856 eye video frames from 22 participants. They found that, despite improvements in general pupil detection accuracy, the algorithms still yield unsatisfactory pupil detection rates under real-world conditions, with eye make-up causing issues in particular (60% of the data for participants wearing eye make-up yields no detection). In turn, inaccurate pupil detection and data loss dramatically affect eye tracking metrics [9]. Whereas inaccurate pupil detection reduces dwell time (total gaze duration in one area of interest from entry to exit), failure to detect the pupil reduces the number of fixations and increases fixation duration [9]. Recently, Fuhl et al. introduced a new pupil detection algorithm named *ElSe* [6] that outperforms other current state-of-the-art approaches (*Swirski*, *ExCuSe*, *Pupil-Labs*, *Starburst* [40] and *Set* [11]) in an evaluation study [7] that used a large-scale composite dataset of previously annotated images (from [36], [6], [34], and [5]). However, while *ElSe* slightly improves on the performance, it cannot yet robustly detect pupil positions in the presence of reflections, poor illumination conditions, or eye make-up (see [7] for detailed results).

Because of the limitations of automated methods, outdoor studies are often avoided, and participants wearing spectacles, eye make-up, or who display Ptosis are commonly excluded. This leads to significant limitations and constraints in how and where eye tracking can be deployed.

Self-reporting based eye tracking

Studies have proposed alternative methods for determining gaze directions without the use of eye trackers. Rudoy et al. [24] developed a self-reporting method to collect gaze direction data from online workers. Workers were asked to watch a video followed by a grid screen with unique codes, which was briefly displayed at the end of each video. Workers were then asked to enter the code they saw most

clearly to indicate their last gaze direction. Although this approach collects the last gaze direction, it fails to collect the direction of first gaze and all other gazes over the period of watching the stimuli. Similarly, Cheng et al. [2] developed a self-reporting gaze direction method based on mouse clicks. Unlike [24], Cheng's approach collects the first and last gaze directions as well as other gaze directions (in between) from online workers. In Cheng's study workers were instructed to look at a static image followed by a 9×9 grid image. The workers are required to memorize the sequence in which they shifted their sight (gaze) from one location on the viewed image to another until the grid is displayed. Workers are then required to recall the locations and sequence of their gaze, and click the relevant grid cell.

Although the two studies report comparable results to those obtained from conventional eye tracking techniques, they suffer from a number of drawbacks, including: i) intrusiveness and full dependence on participants to self-report; ii) an increase in cognitive load that could influence participant responses; iii) a dependence on participants' memory, especially when recalling all gazes; iv) restricted use of on-screen applications only; and v) a lack of other important eye tracking features (e.g. fixation durations and saccades).

As such, robust, unobtrusive and pervasive real-world eye tracking methods remain an unresolved challenge. Since self-reporting methods suffer number of substantial drawbacks, and automated pupil detection algorithms are insufficient in real-world settings, our approach proposes a workaround solution by harnessing the crowd.

Crowdsourcing-based systems

Previous literature has explored the potential for crowdsourcing to supplement automated algorithms when they are found insufficient. Studies have shown that workers do remarkably well in visual tasks that involve recognizing and identifying objects. For instance, Su et al. [32] used crowdsourcing to generate quality image annotations (e.g. fitting a bounding box around each bottle in an image) for over one million images that could be used for training automated object detection systems. Su et al. reported that the crowd successfully annotated 97.9% images with a high accuracy of 99.2% [32]. Similarly, Hipp et al. harnessed the crowd to annotate images from publicly available webcams in two road intersections to outline cyclists, pedestrians and vehicles [8]. They report a high inter class correlation (ICC) between workers, equivalent to the ICC of two trained researchers who completed the same annotation tasks.

Such findings highlight the potential of utilizing the crowd to localize the eye pupil. However, unlike the latter two studies, which focused on counting a target in an image or fitting a bounding box around it, our study focuses on accurate pupil center localization in images in noisy real-world settings (i.e. images could be blurry, or may contain high light reflections making the pupil more challenging to find). While crowdsourcing therefore appears to be a potential candidate to supplement eye tracking pupil detection algorithms, the high number of eye tracking images to be crowdsourced, and the high accuracy level required to localize the pupil center, as well as the associated processing time and costs are major challenges that require further study.

We address these challenges by utilizing frame selection methods to slice out highly similar frames, designing localization and labeling crowd tasks, and introducing a quality assurance method based on self-validation and refinement. The solution is evaluated against the LPW dataset and we report measures for localization accuracy, robustness, and costs.

METHOD

We extend recent work that has explored the use of the crowd in object labeling [8,25,32] and face recognition [31], and the collection of [41]—as well as the self-report on [2,24]—eye tracking data. Unlike Rudoy et al. [24] and Cheng et al. [2], *CrowdEyes* uses a conventional mobile head-mounted eye tracker and requires neither self-reporting (i.e. participants indicate where they gazed) nor user interference for data collection (i.e. participants complete some tasks in order to find their gaze positions). Our system extends the *Pupil* open-source eye tracking platform [12] and consists of two main components: i) a head-mounted mobile eye-tracker based on the *Pupil* open-source platform; and ii) a set of crowd tasks by which both pupil and calibration target (i.e. finger thumbnail) localization can be realized with very high reliability. Crowd workers were recruited from the commercial crowdsourcing platform CrowdFlower[1]. The robustness and accuracy of our approach was evaluated by leveraging the open-source LPW dataset [36] of heterogeneous head-mounted mobile eye tracker recorded under natural (indoor and outdoor) conditions. The results were then compared to [36]'s reported measures of five state-of-the-art algorithms (*Swirski* [34], ExCuSe [5], *Isophete* [37], *Gradient* [35] and *Pupil-Labs* [12]). We also establish and demonstrate a novel approach to crowdsourcing quality control based on a worker response validation and refinement cycle. We further demonstrate the potential for *CrowdEyes* to be extended to include crowd data analysis tasks that are traditionally very time-consuming for researchers; in this case the annotation and labeling of fixations. As such our contribution is to propose and demonstrate crowdsourcing-based approaches for cost-effective, robust, accurate and extensible approaches to pervasive mobile eye tracking.

CROWDEYES DESIGN CONSIDERATIONS

There are two main components in the design of *CrowdEyes*. The first is the use of existing eye image capture hardware and software. The second is the crowd task design, including accommodating crowdsourcing

[1] www.crowdflower.com

platform constraints, data types, and response quality. In this section we highlight the key factors that have shaped the design of *CrowdEyes*.

The eye tracker

Hardware

Currently, commercial mobile eye tracking systems are expensive. Costs range from US $10,000 to $30,000 [2]. At the same time, it is possible to produce DIY head-mounted eye trackers to run by an open-source eye tracking platforms. Since mobile devices lack the support of multiple concurrent camera captures (eye and world), and off-the-shelf portable PCs are not sufficiently powerful to accommodate all eye tracking requirements (e.g. concurrent camera capture, pupil detection and gaze mapping), a workaround is required to provide robust DIY mobile eye tracking.

Software and real-time performance

To reach a wider range of users, it is a common practice to improve an existing well-established and widely utilized platform. Thus, we have utilized the *Pupil* platform [12]. Its open source nature enabled us to implement modifications and to integrate new features, such as the proposed crowdsourcing pipeline for localizing pupil and calibration targets, and for labeling what a person gazed or fixated on. Since eye tracking detection and gaze mapping are computationally intensive, all features other than recordings were turned off during the recording sessions, allowing the use of low-cost pocket PCs.

Crowd tasks

Data volume

During the calibration and recording processes both world (from calibration process only) and eye frames must be crowdsourced to locate the calibration target and the center of pupil respectively. If run by brute-force, this results in an enormous number of images to crowdsource (e.g. one hour of eye tracking yields 108,000 eye images using a 30Hz camera). However, at such high sampling rates, camera frames contain redundant information where the target (pupil or calibration marker) has not moved significantly. To keep final running costs to a minimum, redundant data must be identified and excluded before crowdsourcing.

Presentation

CrowdEyes proposes two tasks: one to localize the target center (pupil or calibration target) and another to validate and refine rejected crowd submissions. The target localization task must be designed to avoid increasing cognitive workload that could impact completion time and decisions. Thus, localization tasks should i) minimize visual search for the targets across the presented images, and ii) reduce page scrolling and mouse movements from one image to another. On the other hand, the validation and refinement task (for rejected submissions) must be designed to allow for a quick overview of all workers' annotated images and easy access to those that require refinement.

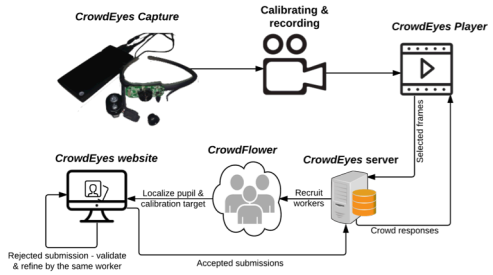


Figure 2. *CrowdEyes* system architecture.

Quality vs. costs

While accurate pupil localization is essential, low data processing costs are also highly desirable. Existing crowdsourcing platforms (e.g. CrowdFlower) provide built-in quality control measures, such as test question injection, and multiple judgments aggregation. Since the multiple judgments aggregation approach increases the final costs, and workers may become aware of the test questions, additional quality measures are required. Moreover, such platforms provide a facility to either remove workers with quality responses lower than a threshold or accept their responses regardless. For *CrowdEyes*, where fine center localization accuracy is a necessity, it would be expensive and unfair to remove workers who spent time and effort completing the tasks but did not achieve high accuracy at the first attempt. Furthermore, increasing standard quality measures and removing workers not meeting the minimum quality standards from the first try will also incur additional costs. Since crowdsourcing platforms give little control over the task's pipeline and quality measures, *CrowdEyes* instead recruits workers from CrowdFlower to complete tasks on an external website. Workers who do not meet the minimum quality measures receive extra opportunities to validate and refine their entries before receiving payment.

THE CROWDEYES SYSTEM

CrowdEyes is composed of: (i) the *Pupil* open source head-mounted eye tracker, comprising a 3D printed frame fitted with two low cost off-the-shelf web cameras (30Hz) to capture the eye and world scene (Figure 2); (ii) portable video capture and processing hardware in the form of a portable pocket PC (Figure 2) running Ubuntu 16.04 and a Bluetooth remote button; (iii) software plugins that link eye tracking software to a crowdsourcing server; and (iv) the crowdsourcing server. The total cost of construction of the eye tracker is approximately US $270 (not including crowdsourcing costs).

The *CrowdEyes* capture component (*Capture*) and player component (*Player*) are written in Python to extend the open-source *Pupil* platform. *Capture* is a lightweight eye and world scene video capture plugin. It disables *Pupil's* functionalities (i.e. runtime detection processes) other than video capture to function with the hardware limitations of current pocket PCs, and saves information about the start and end time of each calibration procedure. *Player* is the plugin that processes *CrowdEyes* captured data offline by

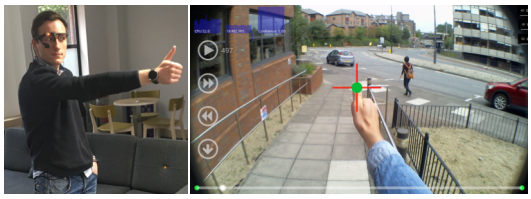


Figure 3. Calibrating while looking at the thumbnail (left), and a player screenshot with overlaid crosshair gaze position (right).

harnessing the crowd. Post recording, *Player* communicates with the crowdsourcing server to: i) delegate pupil and calibration target localization; and (optionally) ii) delegate the labeling of the detected fixations as a crowd annotation task. Using *Pupil's* open source software is crucial to our system. Whereas the *CrowdEyes Player* completes the localization process, the underlying *Pupil* software enables instant access to standard eye tracking functionalities, such as gaze mapping, saccades, as well as fixation positions and durations.

The crowdsourcing server manages the assignment and quality of pupil and calibration target localization as well as fixations labeling. It consists of three components: i) an online web service that mediates between *Player* and CrowdFlower and recruits and manages workers; ii) a web application where workers complete the tasks; and iii) a database server storing the responses gathered from the crowd.

DATA CAPTURE

Recording begins with a user-controlled calibration process that is initiated by powering-up the processing unit. Our early trials showed that computer vision, in real-world scenarios (e.g. outdoor), not only fails to detect the center of pupil but the machine-known calibration target too. Unlike computer vision, the nature of *CrowdEyes* means that any object that can be unambiguously identified by workers can be used as the calibration target. Consequently, the wearer can perform calibration based on features in the environment, such as the handle on a door, or (conveniently) their own thumbnail (Figure 3). For example, a wearer can perform calibration by looking at the nail of her thumb while moving her head (thumb-static), or vice versa, moving her thumb keeping her head static (head-static), such that the target occupies different positions in her visual field. The only constraint is the requirement for a short pause between each movement (as in a typical 9-point calibration method) to allow for the collection of a sufficient number of calibration samples. The calibration process takes on average one minute depending on the wearer and how many pauses (points) they cover. The wearer marks the end of calibration with a click of the remote button. While recording, our system imposes no other constraints. *CrowdEyes* enables the wearer to wear their spectacles, contact lenses, and eye make-up, and to record under any illumination level and under other uncontrolled real-world conditions. To stop recording the wearer clicks the remote button once more, which powers off the processing unit.

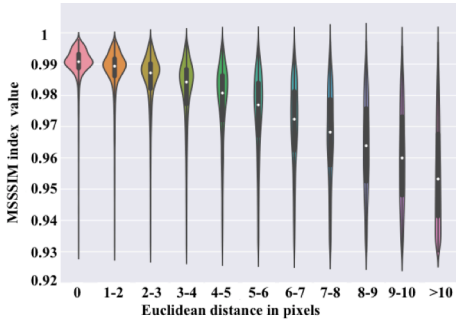


Figure 4. MSSSIM distribution.

PUPIL AND CALIBRATION TARGET LOCALIZATION

The post-hoc localization of the center of pupil and the center of the calibration target (using *CrowdEyes Player* after the recording session is complete) proceeds in three steps: i) frame selection; ii) per frame pupil and calibration target localization; and iii) gaze mapping.

Step1: Frame selection

The *CrowdEyes Player* plugin separates recordings into calibration and post-calibration recording sessions (based on the remote button markers), and decomposes the videos into single frames. Raw decomposition produces a large number of frames. Following [14,26], *CrowdEyes* identifies and groups similar frames (e.g. where the pupil has not moved significantly) and selects one of these for analysis by the crowd. Whereas [14] is usually deployed in almost-static environments, and a similarity check is performed periodically (every n-minutes) on a cropped part of a frame, *CrowdEyes* records and searches for a rapidly moving pupil in changing environment (e.g. lighting reflections). As such, *CrowdEyes* continuously checks all sequential frames for similarities. In contrast to [26], which looks for high significance differences between frames to summarize a video clip, *CrowdEyes* looks for minor changes to the pupil position. And unlike both [14,26], *CrowdEyes* uses a *multi-scale structural similarity index* (MSSSIM) [39] for sequential frames, giving more weight to changes in pupil position than lighting reflections and other irrelevant noise factors.

MSSSIM values range from 0.0 to 1.0, where a value of 1 signifies an identical pair of frames. However, MSSSIM requires sufficient processing power and time to compare thousands of eye tracking pupil-frames with each other. To simplify and speed up this procedure, *Player* blurs and converts frames to gray scale and resizes them down from 640×480px to 160×120px prior to the similarity check. Sequential frames with MSSSIM values >= 0.98 are clustered and the first frame (by MSSSIM value) is added to the crowd job list. The violin plots in Figure 4 show estimates of the density and distributions of MSSSIM values for several intervals of pixel distances between the center of a pupil in consecutive frames using the LPW

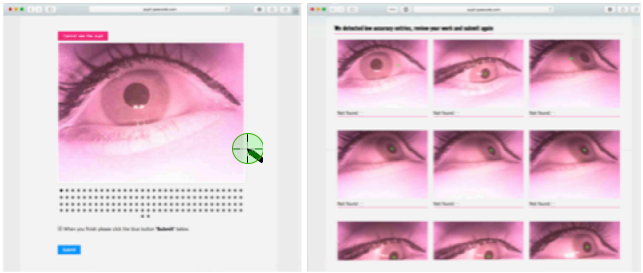


Figure 5. *CrowdEyes* web pages to localize pupil (left) as well as validate and refine workers' low accurate responses (right). The green-circled crosshair is the used cursor.

dataset [36]. The plots suggest the highest density between consecutive frames is when the MSSSIM is greater than 0.985 with no distance differences. Moreover, most of the data with MSSSIM >= 0.98 is no farther than 4 to 5px away from the pupil center of the comparison image. Once frames are selected, *Player* prepares the localization job—a set of images and the associated crowd task description and configuration (i.e. pupil or calibration target localization, payment in cents, number of judgments)—and then submits them to the server for processing by the crowd.

Step 2: Pupil and calibration target localization

Upon receiving the job list, the server recruits workers from CrowdFlower to complete the tasks concurrently on the *CrowdEyes* website. Each worker is instructed to identify either the center of the pupil or the center of the calibration target for 130 sequential images (including 30 gold standard reference images) (640×480px) by clicking on the corresponding point in the image (Figure 5-left). To help workers visually identify the closest point to the center, the default mouse cursor is replaced with a customized crosshair pointer surrounded by a green circle (Figure 5-left). To address the presentation design challenges for locating the target center of many images we resort to image sliders. Only one image at a time is presented to a worker to locate the target's center (Figure 5), then once a worker performs a click, the next image will be presented to locate the next target center and so on.

Quality throughput

Whereas eye tracking requires highly accurate pupil localization, workers are usually after maximum monetary compensation. Hence, workers tend to complete many tasks as fast as possible to increase their daily income, which results in unintentional mistakes. However, with aggressive quality measures the chances of blocking workers unintentionally are high, resulting in unfairness towards workers and additional expenses for requesters. Thus, *CrowdEyes* uses three quality control methods to ensure high quality responses, low costs and fair payment:

Injecting gold standard reference images: We employ this common crowd quality control method, injecting subject-known-center images (eye or calibration frame for relevant localization task) for which a worker must achieve an accuracy (Euclidean distance from the true pupil center) of

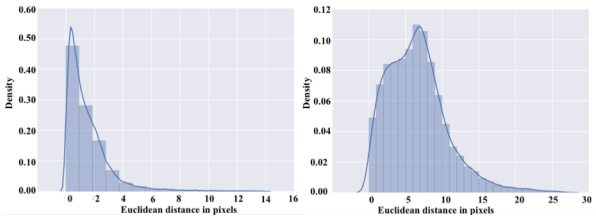


Figure 6. The density of Euclidean distance between consecutive frames (left), and consecutive MSSSIM-selected frames (right).

less than 10 pixels. Moreover, to minimize crowdsourcing costs, instead of increasing the number of judgments per task, *CrowdEyes* builds on a single judgment but increases the test data percentage. Each task contains 30% ground truth sequential images selected randomly from our manually annotated images pool. The percentage of test data is purposefully high so workers cannot identify test data among the others, and to compensate the single judgment per task.

Euclidean distance between two sequential clicks: If the pupil moves rapidly, its center shifts gradually in consecutive frames. Thus, if we estimate the farthest Euclidean distance between two consecutive frames, we can use it as a quality measure to prevent random and robot responses as well as to detect unintentional false responses. Using the LPW dataset [36] we found the farthest distance between two consecutive frames to be under 15px (Figure 6-left), while under 30px between two consecutive MSSSIM-selected frames (Figure 6-right). As such, a worker fails to meet this quality measure when the Euclidean distance between two clicks on two consecutive frames was farther than 15px (all frames task) or 30px (MSSSIM-selected frames task). For example, in an MSSSIM-selected frame job, an entry is rejected if a worker identifies pixel position (325, 230) and for the following frame identifies pixel position (290, 230)—a Euclidean distance of over 30px.

*Time spent: S*ince workers on crowdsourcing platforms are usually low-paid, and tasks are assigned on a first-come-first-served basis, workers often multitask (sign-up for, and undertake, multiple crowd tasks at the same time). Thus, each worker is instructed to complete the localization task within 10 minutes (three times the average completion time) before it is reassigned to the next available worker. Late and inactive workers whose job has been reassigned to others lose their session and receive no payment.

Entry validation and refinement: Unlike traditional crowdsourcing strategies to expel workers with low quality responses [28] without compensation, or simply accept all work regardless of quality, *CrowdEyes* enables workers to validate and refine their own entries. Where workers fail the quality tests they are given the option to refine their entries and submit again. The refinement stage may be completed multiple times until responses satisfy *CrowdEyes* quality

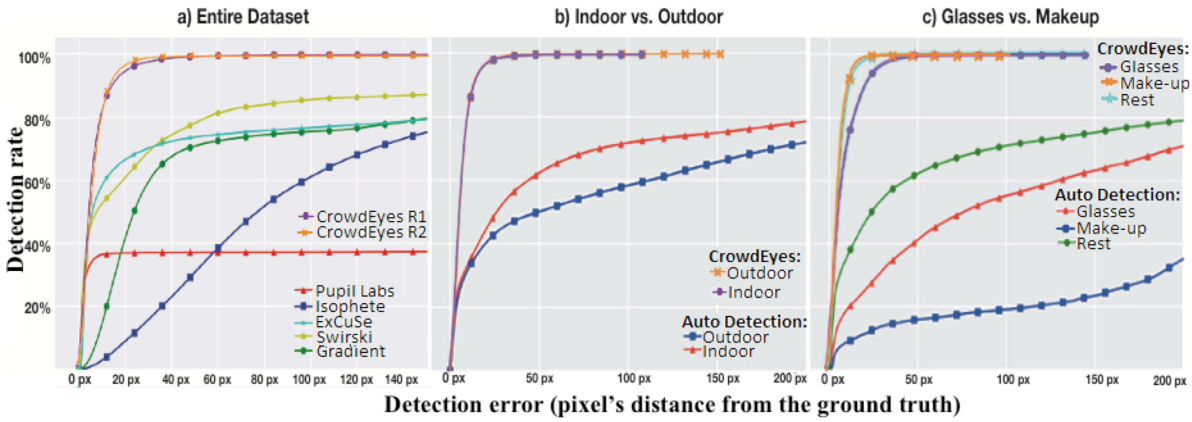


Figure 7. Cumulative distribution of the mean error: a) comparison of *CrowdEyes* method (run 1 (R1) and run 2 (R2) without and with MSSSIM frame selection respectively) and 5 common algorithms (adapted from [36]**); b) comparison of *CrowdEyes* method and automatic detection using frames collected indoors and outdoors; c) comparison of *CrowdEyes* method and automatic detection using frames representing glasses and eye make up**

measures, or the worker gives up. If a worker gives up, all their entries will be rejected and they will not receive a payment. Workers are given 5 extra minutes every time they are asked to improve their responses before the task is reassigned to the next available worker. As such, *CrowdEyes* redirects the worker to the refinement page where her entries are overlaid on the task images. Images are presented in a grid and the worker is requested to validate all entries so they are as close to pupil center as possible and improved accordingly (see Figure 5-right).

Recruitment and Payment

For every eye tracking job, *CrowdEyes* creates a job recruitment page on CrowdFlower. The recruitment page contains a link to *CrowdEyes* tasks website, a text field to enter the payment redeem code, and a client-side script to validate payment with the *CrowdEyes* server. Since *CrowdEyes* recruits external workers from CrowdFlower to complete tasks on the dedicated *CrowdEyes* website, *CrowdEyes* must issue a payment code for workers to redeem their payment on CrowdFlower. To prevent workers from entering the same code twice or sharing it with other workers, we issue a unique payment code assigned to that particular worker. As soon as the code is entered on the CrowdFlower job page, our client-side validation script communicates with the *CrowdEyes* server for approval.

Step 3: Gaze mapping

The *Player* plugin contains a feature for checking the crowd job completion status and to retrieve the crowd responses when ready. Once received, the player component identifies outliers in the results for which it compensates using the calculated mean of the preceding and following frames. Then, using standard *Pupil* functions [12], *Player* calculates gaze positions, saccades, and detects fixations. At this point fixations are not labeled but the user can review recordings on which gaze, saccades and fixations are overlaid.

Step 4: Labeling fixations (Optional)

Eye tracking recordings yield fixations in which eyes are relatively static while looking at a specific location for duration of time. Each fixation has a corresponding set of world scene frames (e.g. a detected 235ms fixation captured by 30Hz camera is composed of 7 frames). *Player* selects the middle (temporally) frame out of each fixation set, eliminating repetitive frames [21], and sends all selected frames to the server to be labeled. The crowd task requires workers to answer questions related to fixations and the surrounding area in each image (e.g. categorize or describe objects being looked at) in a maximum of 10 world scene frames, on each of which a crosshair has been overlaid (corresponding to the fixation point). Upon the completion of the crowd labeling tasks, *Player* retrieves the aggregated crowd responses and overlays them on the recording clip to appear near the fixation's crosshair (see Figure 3-right). *Player* also saves the aggregated results and their relevant timestamps in a spreadsheet.

EVALUATION

We took a two-stage approach to evaluate the *CrowdEyes* solution. First we aimed to ensure that the crowdsourcing approach to pupil localization was sufficient, thus we tested our method using a large-scale open dataset and compared the outcomes with those of existing measures [36]. Second, we evaluated the entire solution in a real-world scenario, moving through the entire pipeline from pupil calibration, over data capture, to analysis.

Stage 1: Evaluation of pupil localization

Methods & Procedure

The *CrowdEyes* analysis pipeline (i.e. pupil localization) was assessed for accuracy, robustness, and cost, using an open benchmark dataset [36] of 66 heterogeneous recordings of 22 participants (5 different nationalities) totaling 130,856 frames captured in unconstrained environments (22 minutes of footage captured at 95fps). This dataset was chosen since it includes four distinct and challenging conditions: users with spectacles; users wearing eye make-up; and outdoor as well as indoor scenes with mixed light. The 66 recordings were crowdsourced twice: i) all frames without MSSSIM frame selection in the first run

(R1), and ii) with MSSSIM frame selection in the second run (R2), and compared to [36]'s reported measures of the five state-of-the-art algorithms.

Results and analysis

1. Accuracy and robustness

CrowdEyes in R1 and R2 significantly outperformed all five algorithms for *cumulative distribution* (*CD*) of the mean error in pixels on the LPW dataset. Figure 7a indicates that *CrowdEyes* localized the pupil center for 100% of frames (in all conditions) in both R1 and R2 with a detection error (pixel's distance from the ground truth) less than 10px for 80% and less than 20px for 97% data. To the contrary, the best two evaluated algorithms, *Swirski* and *ExCuSe*, failed to detect 15-20% frames (in all conditions) and yield a detection error over 20px for more than 35% data (and over 100px for more than 20%). Moreover, *CrowdEyes* (R1 and R2 together) demonstrated incomparable results to localize pupil center under challenging conditions. *CrowdEyes* yields a *CD* mean detection error under 25px for 99% indoors and outdoors data (Figure 7b) despite eye make-up (Figure 7c). However, it is notable that workers responses were less accurate with data of participants wearing spectacles and resulted in *CD* mean detection error under 25px for 90% of the data (Figure 7c). This was mainly due to the pupil being (partially) occluded from the eye camera field of view by the spectacle's frame. Nevertheless, unlike *CrowdEyes* all five algorithms yield a *CD* mean detection error over 50px for 40% of indoors and 50% of outdoors data, and over 100px of more than 80% data when participants are wearing eye make-up—not to mention more than 60% of data for participants wearing eye make-up is undetected. These results also suggest using the MSSSIM index to elicit redundant frames in R2 not only reduces costs but also maintains comparable levels of accuracy as R1.

	R1	R2
Frames	130,856	27,230
Micro-tasks	1309	273
Workers (no tasks completed)	1375 (39)	305 (17)
Refined (success rate)	120 (77.5%)	59 (74.6%)
Cost	$523	$109
Time Mean (STD)	175s (56s)	179s (63)

Table 1. R1: Crowdsourcing all video frames; R2 with frame selection using MSSSIM, for 66 recordings with 95Hz cameras (about 23 minutes).

2. Time and costs

The total number of micro-tasks for R1 was 1309 (130856 frames). 1375 micro-tasks were assigned in total because for 39 micro-task assignments workers failed to complete any task and a further 27 micro-tasks workers either gave up on refining their entries or were timed out. 93 workers successfully refined their entries for micro-tasks after one or more refinement trials. For R2, the application of

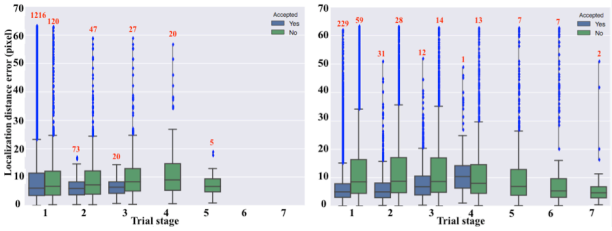


Figure 8. Refinement trials vs. workers' localizations distance error in pixels for accepted (blue box) and rejected (green box) submissions (total in red) from first run R1 (left) and second run R2 (right).

MSSSIM resulted in a reduction in the total number of frames by approximately 80%, hence an 80% reduction in the costs of crowdwork. In total 305 micro-tasks were assigned, since for 17 micro-task assignments workers failed to complete any tasks, in a further 15 micro-tasks workers either gave up on refining their entries or were timed out, and 44 workers successfully refined their entries for micro-tasks after one or more refinement trials. The mean time taken to complete a task in R1 as well as R2 was just under 3 minutes. In total, R1 took 57 minutes to complete compared to 26 minutes for R2. We paid US $0.4 per worker per task (100 frames plus 30 gold injected frames), a pay rate equivalent to the UK minimum wage. The total cost for crowdsourcing R1 (all frames) was US $523 (US $22.7 per 95Hz eye tracking minute), while it was US $109 (US $4.7 per 95Hz eye tracking minute) for R2 (MSSSIM-frame selection), see Table 1. This is approximately US $7.2 per eye tracking minute (all frames 30Hz camera) or US $1.4 per eye tracking minute (MSSSIM selected frames 30Hz camera). Hence, *CrowdEyes* enables us to capture accurate and robust eye tracking sessions for as little as US $87 per hour of data using 30Hz sampling rate cameras.

3. Refinement

Figure 8 illustrates the localization distance error distribution (from the true center) for the accepted (blue box) and rejected (green box) submissions (total submissions in red) during the target localization trial (trial 1) and the refining trials (trial 2 and above). Responses are accepted when a submission meets all quality measures, or rejected when it fails one or more quality measures. Figure 8 left (R1) trial 1 shows that response distribution for rejected submissions is equivalent to the accepted ones, indicating workers may fail one or more quality measures despite their overall good responses. From here, 73 workers in trial 2 and another 20 workers in trial 3 successfully refined their responses and significantly achieved an even better distance error distribution than that accepted in trial 1. It is worth mentioning that outliers are almost eliminated in the accepted refinement trials. However, after the third trial remaining workers either timed out or gave up on refining, or kept on failing one or more quality measures. Similarly, Figure 8 right (R2) illustrates the significant improvement to submissions in the refinement trials. This suggests that trusting workers to validate and refine their

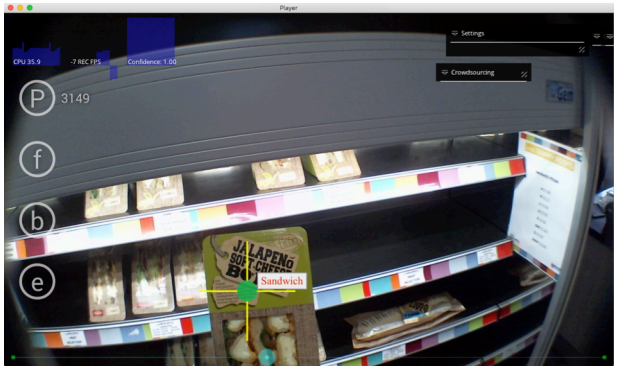


Figure 9 *CrowdEyes Player* plugin integrated into the open-source *Pupil Player* software to show the labeled fixations.

responses results in significant quality improvement, offers workers fair compensation for their effort and time, and keeps costs to a minimum. As a result, 137 (out of 179) workers (across R1 and R2) successfully completed their refinement tasks and guaranteed their compensation. Finally, it appears that the refinement quality method also motivates workers to visit our job again. Approximately 33% and 51% of workers who were accepted after the refinement trials in R1 and R2 respectively returned to complete more tasks.

Stage 2: Applying *CrowdEyes* within a real-life scenario/

Methods & procedure

Here we illustrate the utility of *CrowdEyes*, and its extensibility to accommodate further crowdsourced tasks. This includes fixation labeling, which can present *CrowdEyes* users with summaries of where users focused on with their gaze, a common interest in the analysis of eye tracking data. We recruited 8 participants (6 male and 2 female, all University employees or students, four with spectacles, and one with eye make-up) to use *CrowdEyes* to capture what they pay attention to with their gaze when purchasing food in their workplace cafeteria. Participants were instructed how to carry out the initial calibration procedure. We asked participants to use their thumbnail instead of the standard calibration marker, as our initial trials have shown automated detection of known markers to be problematic in light-filled and object crowded environments. As per the traditional 9-point calibration method, the predefined points are relative to the world camera field of view and user's head position. Thus, by using the thumbnail, the predefined points are also appearing in different positions in the world camera's field of view as the user moves her hand (or head) (see Figure 3). All participants chose to calibrate with a stationary thumb (moving their head)—looking at their thumbnail, covering the upper, middle and lower rows of their visual field and pausing three times on each row. Three participants chose to calibrate the eye tracker outdoors before commencing their purchase indoor. Participants were instructed to activate and calibrate the eye-tracker before entering the cafeteria, purchase their lunch, and turn off the recording on completion of their purchase. The overall recording time

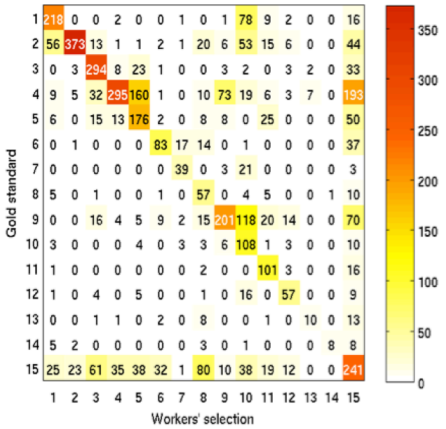


Figure 10. Confusion matrix illustrating the agreement between the categories selected by the crowd workers and the correct (gold standard) categories.

(all footage) was 28:56 minutes (shortest=1:50; longest=5:37; average=3:36) using 30Hz cameras. The total calibration time was 08:25 minutes (shortest=00:42; longest=01:23; average =01:03), and the total number calibration world frames was 15,150.

Results & Analysis

1. Time and costs

The total number of MSSSIM-selected eye frames to crowdsource pupil localizations was 10,104 out of 52,093, which is a reduction by approximately 81%. This resulted in 102 micro-tasks completed successfully by 102 workers; nine workers had to refine their entries before being accepted. Additionally, calibration world frames were crowdsourced to localize the calibration target (in this study the tip of the finger thumb). The application of MSSSIM to select world frames from the calibration process resulted in a reduction by 42% (8,723 frames, 88 micro-tasks). This was not as effective as applying it to eye frames, which may be due to the higher noise factors (e.g. mixed light, many objects in the field of view) in the world frames compared to eye frames. However, the overall cost for pupil localization was US $41 (US $1.4 per minute), plus US $35 to localize the calibration target in all recordings (US $4.4 per calibration session).

The total number of fixations was 1406 (excluding fixations during calibration), which resulted in 141 micro-tasks (each micro-task consisting of a maximum 10 world scene frames and 2 gold injected frames, judged by 3 workers). We recruited 456 workers from CrowdFlower to complete the tasks on *CrowdEyes* website. Among them 21 quit the tasks too early; 12 others timed out or gave up on refining their entries; and for 49 micro-tasks workers had to refine their answers to meet with the quality standard. Workers were instructed to categorize the object being fixated upon (identified by a crosshair) using a provided list of categories. The categories used were: 'Man', 'Woman', 'Group of people', 'Drink', 'Sandwich', 'Chocolate bar, crisps, chips, biscuits', 'Cash register', 'Display Screen',

'Table or chair', 'Sign, post or advertisement', 'Wall', 'Floor', 'Gate or Door', 'Fruit', 'Other'. These categories were given after the research team went through the recordings and identified every possible item or object that the wearer could have fixated on. The mean time taken of all micro-tasks was ~114 seconds (STD=61 seconds). Successful workers were paid US $0.3 per micro-task (US $127 per full job).

2. Accuracy and robustness

Since this paper already demonstrated the accuracy of localizing the center of pupil above, in this section we focus on evaluating crowd responses related to just the fixations labeling task. Since the categories used for labeling fixations are nominal and judged by more than two workers, Fleiss' kappa is used to measure the inter-rater reliability (IRR) between workers. Despite the simple quality control measure employed, the IRR results suggest substantial levels of agreement between workers, corresponding to a Fleiss' kappa of 0.6671. To accumulate a gold standard, we have manually annotated the detected fixations prior to crowdsourcing, and used 10% of images for the injected gold standard quality measure. The crowd results were then compared with our gold standard annotation. The confusion matrix in Figure 10 illustrates the agreement between the categories selected by the workers and the gold standard categories. The values on the diagonal correspond to cases where the targets were recognized by the workers as belonging to the correct categories. The unweighted Cohen's kappa coefficient computed from this matrix is 0.49. This moderate level of agreement is due to difficulties in distinguishing between some of the categories, which results in some high values outside of the diagonal in Figure 10. This could be due to the limited-training workers received, beside the quality of the captured image, the distance of the object being fixated on from the camera, or the object being unknown to workers. For example, category 15 ("Other") was used whenever the workers didn't recognize the object behind the crosshair. Removing this one category would result in a kappa of 0.61 (substantial agreement).

Finally, the processed data is presented using *Pupil Player* software and *CrowdEyes Player* plugin. Figure 9 presents a selected frame from the lunch purchase process with the crowd-labeled fixation (Sandwich).

DISCUSSION

CrowdEyes demonstrates that crowdsourcing (human-computation) can be employed to improve data processing and analysis for wearable mobile eye trackers. Our studies deliver robust comparative findings, showing high pupil tracking accuracy and suggest that fixation labeling can also be automated to deliver reliable and telling outcomes. While employing workers for these tasks does come at a cost, projections including broader worker audiences and a tolerable reduction in key frames that are sent out for manual detection suggest that eye tracking data analysis with *CrowdEyes* can be efficient and scale to a low per minute cost, while delivering a level of quality that is unparalleled by purely computational approaches. As evidenced by our findings, giving workers further opportunity to validate and refine their entries yielded better levels of performance, higher rates of task completion, more compensation awarded to workers and, importantly, more workers revisiting the job.

Whereas the self-reporting gaze recall methods [2,24] require no other special hardware than a display screen, *CrowdEyes* requires a head-mounted video-based eye tracker. In turn, *CrowdEyes* expands the *Pupil* platform, adding a human-computation plugin, and using a pocket PC, two off-the-shelf webcams and a 3D printed head-mounted frame—low-cost and hackable. However, unlike [24] and [2], which must be performed on screen while workers complete number of memory-dependent tasks to recall gaze positions, *CrowdEyes* enables robust as well as mobile eye tracking under real-world conditions, with few constraints regarding locations, lighting conditions, or eyewear. This means that eye tracking can be used, for instance, to efficiently evaluate outdoor activities (e.g. visual attention for cyclists when cycling on or off road) and technologies (e.g. the impact of using mobile phones on situational awareness during a walk). Moreover, it can also be used as a lifelogging tool that video captures and labels the surrounding area as well as the wearer gaze and fixations, adding more depth to lifelogging captured data. As a result, *CrowdEyes* could eventually be used to drive recommender systems based on what a wearer looked at.

Limitations and Future Work

While the evaluations presented here were designed to include realistic use cases, the approach does require ecological validation, which is especially relevant to gauging the value of future applications of the fixation labeling process. The durations of the eye tracking recordings employed in these studies were substantial, but the question of how easily the approach scales to longer duration recordings does require further evaluation, as do considerations related to potential near real-time analysis through further parallelization. Furthermore, the process for pupil localizations partially relies on gold-standard data. It can be argued that it will likely not be necessary to employ novel gold data samples for the analysis of future recordings, since existing gold data frames could simply be reused. The gold standard data itself, however, also poses a limitation on the study. Given that some of Tonsen's dataset was human annotated, there was possibly a bias towards human annotation methods. In addition, images within Tonsen's dataset were captured with a 95Hz camera, whereas *CrowdEyes* only employed a 30Hz camera. Since the workers' localization accuracy is independent of camera frame rate, unlike the costs, we evaluated the localization accuracy and costs of our approach with Tonsen's dataset (95Hz) in Stage 1 compared to costs only in Stage 2 (30Hz). Consequently, we reported the costs difference in

running *CrowdEyes* with 30Hz cameras (~US $85 for localizations) compared to 95Hz cameras (~US $280). However, to reduce the costs, speed up the process and ensure higher labeling agreement, in our future work we will look at training crowd workers and create a pool of trained workers available on demand. Lastly, the promising outlook of improving automated methods through crowdsourced high quality results, e.g. by training modern deep learning networks, certainly warrants further study.

CONCLUSION

In this paper, we have presented the motivation, design and evaluation of *CrowdEyes*, a hybrid eye tracking system that employs crowdsourcing for pupil and calibration target localizations, combined with automatic data processing (e.g. gaze mapping) provided by standard functionalities of the *Pupil* framework. *CrowdEyes* leverages the crowd to provide a robust and reliable mobile eye-tracker that functions under real-world conditions, a feat that has so far remained elusive. The high accuracy of *CrowdEyes* in localizing pupil center highlights the potential it holds for enabling a broad variety of applications beyond those that are available when using regular contemporary eye tracking only. Moreover, in this paper we have presented a novel crowd quality measure, which relies on workers to validate and refine their entries. This method yields more accurate entries, encourages workers to perform better, and prevents honest workers from being rejected or unpaid. The results of this work suggest our approach is robust, accurate, and cost effective.

ACKNOWLEDGEMENTS

This research was funded by EPSRC Grant No: EP/K037366/1 MyPLACE: Mobility and Place for the Age Friendly City. Data supporting this publication is openly available under an 'Open Data Commons Open Database License'. Additional metadata are available at: http://dx.doi.org/10.17634/122839-1. Please contact Newcastle Research Data Service at rdm@ncl.ac.uk for access instructions.

REFERENCES

1. Jose Javier Bengoechea, Arantxa Villanueva, and Rafael Cabeza. 2012. Hybrid eye detection algorithm for outdoor environments. *Proceedings of the 2012 ACM Conference on Ubiquitous Computing - UbiComp '12*, 685. http://doi.org/10.1145/2370216.2370365

2. S Cheng, Z Sun, X Ma, and JL Forlizzi. 2015. Social Eye Tracking: Gaze Recall with Online Crowds. *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, 454–463. Retrieved March 17, 2015 from http://dl.acm.org/citation.cfm?id=2675249

3. Nicholas S. Dalton, Emily Collins, and Paul Marshall. 2015. Display Blindness? Looking Again at the Visibility of Situated Displays using Eye Tracking. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*, 3889–3898. Retrieved from http://dl.acm.org/citation.cfm?id=2702123.2702150&coll=DL&dl=ACM&CFID=699746312&CFTOKEN=62499401

4. Karen M. Evans, Robert A. Jacobs, John A. Tarduno, and Jeff B. Pelz. 2012. Collecting and Analyzing Eye-tracking Data in Outdoor Environments. *Journal of Eye Movement Research*, 1–19. Retrieved January 14, 2015 from http://www.bcs.rochester.edu/people/robbie/Evans-etal_JEMR2012.pdf

5. Wolfgang Fuhl, Thomas Kübler, Katrin Sippel, Wolfgang Rosenstiel, and Enkelejda Kasneci. 2015. Excuse: Robust pupil detection in real-world scenarios. *Proc. CAIP* 9256. http://doi.org/10.1007/978-3-319-23192-1_4

6. Wolfgang Fuhl, Thiago C. Santini, Thomas Kübler, and Enkelejda Kasneci. 2016. ElSe : Ellipse Selection for Robust Pupil Detection in Real-World Environments. *Eye Tracking Research & Applications*, 123–130. http://doi.org/10.1145/2857491.2857505

7. Wolfgang Fuhl, Marc Tonsen, Andreas Bulling, and Enkelejda Kasneci. 2016. Pupil detection in the wild : An evaluation of the state of the art in mobile head-mounted eye tracking. *Machine Vision and Applications*. http://doi.org/10.1007/s00138-016-0776-4

8. J Aaron Hipp, Alicia Manteiga, Amanda Burgess, Abby Stylianou, and Robert Pless. 2015. Cameras and Crowds in Transportation Tracking. *Proceedings of the conference on Wireless Health - WH '15*, 1–8. http://doi.org/10.1145/2811780.2811941

9. Kenneth Holmqvist, M Nyström, and F Mulvey. 2012. Eye tracker data quality: what it is and how to measure it. *Eye Tracking Research & Applications* 1, 212, 45–52. http://doi.org/10.1145/2168556.2168563

10. K.S. Jacob, R.J.K., and Karn. 2003. "Eye tracking in human computer interaction and usability research: Ready to deliver the promises", In The Mind's Eye: Cognitive and Applied Aspects of Eye Movement Research. *The Mind's Eye: Cognitive and Applied Aspects of Eye Movement Research.*, 573–605.

11. Amir-Homayoun Javadi, Zahra Hakimi, Morteza Barati, Vincent Walsh, and Lili Tcheang. 2015. SET: a pupil detection method using sinusoidal approximation. *Frontiers in neuroengineering* 8, April, 4. http://doi.org/10.3389/fneng.2015.00004

12. Moritz Kassner, William Patera, and Andreas Bulling. 2014. Pupil: An Open Source Platform for Pervasive Eye Tracking and Mobile Gaze-based Interaction.

arXiv preprint, 10. http://doi.org/10.1145/2638728.2641695

13. Ami Klin, Warren Jones, Robert Schultz, Fred Volkmar, and Donald Cohen. 2002. Visual fixation patterns during viewing of naturalistic social situations as predictors of social competence in individuals with autism. *Archives of general psychiatry* 59, 9, 809–816. http://doi.org/10.1001/archpsyc.59.9.809
14. Gierad Laput, Walter S Lasecki, Jason Wiese, et al. 2015. Zensors : Adaptive , Rapidly Deployable , Human - Intelligent Sens or Feeds. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, 1935–1944. Retrieved from http://dl.acm.org/citation.cfm?id=2702416
15. Ji Woo Lee, Hwan Heo, and Kang Ryoung Park. 2013. A novel gaze tracking method based on the generation of virtual calibration points. *Sensors (Basel, Switzerland)* 13, 8, 10802–22. http://doi.org/10.3390/s130810802
16. Dongheng Li, Jason Babcock, and Derrick J. Parkhurst. 2006. openEyes: a low-cost head-mounted eye-tracking solution. *Proceedings of the 2006 Symposium on Eye Tracking Research and Applications*, 95–100. http://doi.org/10.1145/1117309.1117350
17. Andrew K Mackenzie and Julie M Harris. 2014. Characterizing Visual Attention During Driving and Non-driving Hazard Perception Tasks in a Simulated Environment. *Proceedings of the Symposium on Eye Tracking Research and Applications* 2011, 127–130. http://doi.org/10.1145/2578153.2578171
18. Päivi Majaranta and Andreas Bulling. 2014. Eye Tracking and Eye-Based Human-Computer Interaction. In *Advances in Physiological Computing*, Stephen H Fairclough and Kiel Gilleade (eds.). Springer London, London, 39–65. http://doi.org/10.1007/978-1-4471-6392-3_3
19. Alessandra Mantuano, Silvia Bernardi, and Federico Rupi. 2016. Cyclist gaze behavior in urban space: An eye-tracking experiment on the bicycle network of Bologna. *Case Studies on Transport Policy*. http://doi.org/http://dx.doi.org/10.1016/j.cstp.2016.06.001
20. Svenja Marx, Gesine Respondek, Maria Stamelou, et al. 2012. Validation of mobile eye-tracking as novel and efficient means for differentiating progressive supranuclear palsy from Parkinson's disease. *Frontiers in behavioral neuroscience* 6, December, 88. http://doi.org/10.3389/fnbeh.2012.00088
21. Susan M. Munn, Leanne Stefano, and Jeff B. Pelz. 2008. Fixation-identification in dynamic scenes. *Proceedings of the 5th symposium on Applied perception in graphics and visualization - APGV '08* 1, 212, 9. http://doi.org/10.1145/1394281.1394287
22. Kimihiko Nakano. 2014. Gaze Measurement to Evaluate Safety in Using Vehicle Navigation Systems. *Proc. IEEE International Conference on Systems, Man, and Cybernetics*, 3972–3977.
23. Christina Ohm, M Mueller, Bernd Ludwig, and Stefan Bienk. 2014. "Where is the Landmark? Eye Tracking Studies in Large-Scale Indoor Environments. *Proceedings of the 2nd International Workshop on Eye Tracking for Spatial Research*, 47–51.
24. Dmitry Rudoy, Dan B Goldman, Eli Shechtman, and Lihi Zelnik-Manor. 2012. Crowdsourcing gaze data collection. *Proceedings Collective Intelligence*. Retrieved January 15, 2015 from http://arxiv.org/abs/1204.3367
25. Bryan C. Russell, Antonio Torralba, Kevin P. Murphy, and William T. Freeman. 2005. LabelMe: A database and web-based tool for image annotation. *International Journal of Computer Vision* 77, 1–3, 157–173. http://doi.org/10.1007/s11263-007-0090-8
26. Sharanjeet Kaur Sandhu and Anupam Agarwal. 2015. Summarizing Videos by Key frame extraction using SSIM and other Visual Features. *Proceedings of the Sixth International Conference on Computer and Communication Technology 2015*, 209–213. http://doi.org/10.1145/2818567.2818607
27. Helmut Schrom-Feiertag, Christoph Schinko, Volker Settgast, and Stefan Seer. 2014. Evaluation of guidance systems in public infrastructures using eye tracking in an immersive virtual environment. *Proceedings of the 2nd International Workshop on Eye Tracking for Spatial Research* 1241, 62–66. Retrieved from http://www.scopus.com/inward/record.url?eid=2-s2.0-84913582404&partnerID=tZOtx3y1
28. M. Six Silberman, Lilly Irani, and Joel Ross. 2010. Ethics and Tactics of Professional Crowdwork. *Xrds* 17, 2, 39–43. http://doi.org/10.1145/1869086.1869100
29. Hardeep Singh, Jagjit Singh Bhatia, and Jasbir Kaur. 2011. Eye tracking based driver fatigue monitoring and warning system. *India International Conference on Power Electronics, IICPE 2010*. http://doi.org/10.1109/IICPE.2011.5728062
30. Hari Singh and Jaswinder Singh. 2012. Human Eye Tracking and Related Issues: A Review. *International Journal of Scientific and Research …* 2, 9, 1–9. Retrieved January 28, 2015 from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.375.5607&rep=rep1&type=pdf
31. Pawan Sinha, Benjamin Balas, Yuri Ostrovsky, and Richard Russell. 2006. Face recognition by humans: Nineteen results all computer vision researchers should know about. *Proceedings of the IEEE* 94, 11, 1948–

1961. http://doi.org/10.1109/JPROC.2006.884093

32. Hao Su, Jia Deng, and Li Fei-fei. 2012. Crowdsourcing Annotations for Visual Object Detection. *Proc. AAAI Human Computation'12*, 40–46.

33. Yusuke Sugano and Andreas Bulling. 2015. Self-Calibrating Head-Mounted Eye Trackers Using Egocentric Visual Saliency. *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, 363–372.

34. Lech Swirski, Andreas Bulling, and Neil Dodgson. 2012. Robust real-time pupil tracking in highly off-axis images. *Proceedings of the Symposium on Eye Tracking Research and Applications*, 1–4. http://doi.org/10.1145/2168556.2168585

35. Fabian Timm and Erhardt Barth. 2011. Accurate eye centre localisation by means of gradients. *Proc. Computer Vision Theory and Applications*, 125–130. http://doi.org/10.5220/0003326101250130

36. Marc Tonsen, Xucong Zhang, Yusuke Sugano, and Andreas Bulling. 2016. Labeled pupils in the wild: A dataset for studying pupil detection in unconstrained environments. *Proc. Eye Tracking Research and Applications*, 139–142. http://doi.org/10.1145/2857491.2857520

37. Roberto Valenti and Theo Gevers. 2012. Accurate eye center location through invariant isocentric patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, 9, 1785–1798. http://doi.org/10.1109/TPAMI.2011.251

38. Arantxa Villanueva and Rafael Cabeza. 2008. Gaze Estimation With One Calibration Point. *Proc. IEEE Transactions on Systems, Man, and Cybernetic* 38, 4, 1123–1138.

39. Zhou Wang, Eero P Simoncelli, and Alan C Bovik. 2003. Multi-scale structural similarity for image quality assessment. *IEEE Asilomar Conference on Signals, Systems and Computers* 2, 9–13. http://doi.org/10.1109/ACSSC.2003.1292216

40. D. Winfield and D.J. Parkhurst. 2005. Starburst: A hybrid algorithm for video-based eye tracking combining feature-based and model-based approaches. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) - Workshops* 3, 79–79. http://doi.org/10.1109/CVPR.2005.531

41. P Xu, KA Ehinger, and Y Zhang. 2015. TurkerGaze: Crowdsourcing Saliency with Webcam based Eye Tracking. *arXiv preprint arXiv: 1504.06755*. http://doi.org/10.1103/PhysRevD.91.123531

42. Chuang-wen You, Nicholas D Lane, Fanglin Chen, et al. 2012. CarSafe App: Alerting Drowsy and Distracted Drivers using Dual Cameras on Smartphones Categories and Subject Descriptors. *Proceeding of the 11th annual international conference on Mobile systems, applications, and services*, 1–14. http://doi.org/10.1145/2462456.2465428