12-2021

iMon: Appearance-based gaze tracking system on mobile devices

Sinh HUYNH

Rajesh Krishna BALAN
Singapore Management University, rajesh@smu.edu.sg

JeongGil KO

Citation

HUYNH, Sinh; BALAN, Rajesh Krishna; and KO, JeongGil. iMon: Appearance-based gaze tracking system on mobile devices. (2021). *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*. 5, (4), 1-26.
Available at: https://ink.library.smu.edu.sg/sis_research/6708

iMon: Appearance-based Gaze Tracking System on Mobile Devices

SINH HUYNH, School of Integrated Technology, Yonsei University, South Korea
RAJESH KRISHNA BALAN, School of Information Systems, Singapore Management University, Singapore
JEONGGIL KO*, School of Integrated Technology, Yonsei University, South Korea

Gaze tracking is a key building block used in many mobile applications including entertainment, personal productivity, accessibility, medical diagnosis, and visual attention monitoring. In this paper, we present *iMon*, an appearance-based gaze tracking system that is both designed for use on mobile phones and has significantly greater accuracy compared to prior state-of-the-art solutions. *iMon* achieves this by comprehensively considering the gaze estimation pipeline and then overcoming three different sources of errors. First, instead of assuming that the user's gaze is fixed to a single 2D coordinate, we construct each gaze label using a probabilistic 2D heatmap gaze representation input to overcome errors caused by microsaccade eye motions that cause the exact gaze point to be uncertain. Second, we design an image enhancement model to refine visual details and remove motion blur effects of input eye images. Finally, we apply a calibration scheme to correct for differences between the perceived and actual gaze points caused by individual Kappa angle differences. With all these improvements, *iMon* achieves a person-independent per-frame tracking error of 1.49 cm (on smartphones) and 1.94 cm (on tablets) when tested with the GazeCapture dataset and 2.01 cm with the TabletGaze dataset. This outperforms the previous state-of-the-art solutions by ~22% to 28%. By averaging multiple per-frame estimations that belong to the same fixation point and applying personal calibration, the tracking error is further reduced to 1.11 cm (smartphones) and 1.59 cm (tablets). Finally, we built implementations that run on an iPhone 12 Pro and show that our mobile implementation of *iMon* can run at up to 60 frames per second – thus making gaze-based control of applications possible.

CCS Concepts: • **Human-centered computing → Ubiquitous and mobile computing systems and tools**.

Additional Key Words and Phrases: Mobile gaze tracking; Appearance-based gaze tracking; Mobile deep learning

ACM Reference Format:
Sinh Huynh, Rajesh Krishna Balan, and JeongGil Ko. 2021. iMon: Appearance-based Gaze Tracking System on Mobile Devices. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 5, 4, Article 161 (December 2021), 26 pages. https://doi.org/10.1145/3494999

1 INTRODUCTION

Gaze tracking has been proposed as a key input method for numerous compelling applications across different domains such as entertainment/games [9, 36], personal productivity [8], human-computer interaction [34, 42, 57], medical diagnosis [10, 12, 14], and behavioral studies [6, 45]. This is because up to 80% of human sensory information is perceived via the visual pathway [21] and thus knowing where the user is currently focusing on within the screen is key for many types of user-driven context-sensitive applications.

[*] Corresponding Author: jeonggil.ko@yonsei.ac.kr

Authors' addresses: Sinh Huynh, School of Integrated Technology, Yonsei University, 50 Yonsei-Ro, Seodaemun-Gu, Seoul, South Korea; Rajesh Krishna Balan, School of Information Systems, Singapore Management University, 80 Stamford Road, Singapore; JeongGil Ko, School of Integrated Technology, Yonsei University, 50 Yonsei-Ro, Seodaemun-Gu, Seoul, South Korea.

2474-9567/2021/12-ART161 $15.00
https://doi.org/10.1145/3494999

Table 1. Person-independent per-frame 2D gaze estimation accuracy (Euclidean error in cm) on the GazeCapture dataset using different techniques from previous studies (top) and ours (bottom).

Study	Description	Phone	Tablet	Average
Krafka et al. (2016) [32]	New model architecture (iTracker) + AlexNet CNN modules	2.04	3.32	2.26
Kannan et al. (2017) [29]	iTracker + AlexNet CNN modules + larger input image size	1.75	-	-
He et al. (2019) [22]	New model architecture (SAGE) + AlexNet CNN modules	1.78	2.72	1.94
Gou et al. (2019) [18]	iTracker + improved CNN modules + new training scheme	1.77	2.66	1.92
Our baseline	SAGE + EfficientNetB3 CNN modules	1.66	2.31	1.77 (+8%)
Improvement (i)	Our baseline + 2D heatmap gaze representation (Sec. 4.1)	1.58	2.07	1.66 (+16%)
Improvement (ii)	Our baseline + eye-region image enhancement (Sec. 4.2)	1.58	2.09	1.67 (+15%)
iMon - full system	Our baseline + (i) + (ii)	**1.49**	**1.94**	**1.57 (+22%)**

Mobile devices, in particular, are an ideal platform for gaze tracking applications as they are the primary computing platform for most users. Unfortunately, even with extensive research efforts, gaze tracking is still far from being a pervasive technology available on all smartphones. Existing mobile gaze tracking solutions either perform inaccurately under real-world settings or require specialized and expensive hardware. In this paper, we present a practical gaze tracking solution that can run entirely on mobile devices without requiring any additional hardware while still achieving best-in-class accuracy.

There are two broad methods of gaze tracking known as i) geometry-based and ii) appearance-based [20] approaches. Geometry-based methods use geometric models of the eyes and specific eye-related features (e.g., iris, pupil) to perform gaze estimation. However, this approach usually requires specialized sensing hardware and a complicated calibration procedure to achieve accurate results.

The second method, known as appearance-based gaze tracking [38, 58], leverages only the features extracted from face and eye images and uses these features to predict the gaze. Recently, convolutional neural network (CNN) models have been used to improve the performance of appearance-based gaze tracking [58] and allowed them to surpass the accuracy of classical feature-based regression models [32, 65]. In addition, appearance-based gaze tracking can be used on nearly all modern commodity smartphones and tablets as all it requires is images, captured non-intrusively, from the front-facing camera.

In this paper, we present *iMon*[1], a mobile gaze tracking system that outperforms the state-of-the-art appearance-based techniques [18, 22, 37]. It does this by considering the entire end-to-end pipeline involved in mobile appearance-based gaze tracking. Previous studies have made improvements on parts of this pipeline, especially the gaze estimation model, but this work argues and shows that the overall performance is impacted not only by the model, but also by carefully engineering and fully exploiting the potential of all pipeline components. In particular, we identify three common sources of error within the pipeline that significantly affect the overall accuracy of the appearance-based gaze tracking systems.

We correct these errors by **(1)** leveraging a novel 2D heatmap-based probabilistic representation of the human gaze to mitigate the errors in ground-truth labels of the gaze point caused by microsaccade eye movements, **(2)** improving the pre-processing pipeline (detect eye regions, enhance visual details and remove motion blur in input images), and **(3)** applying calibration to alleviate the effect of individual differences in Kappa angle, which is the difference between the pupillary and visual axis of a person's eye where larger angles can cause gaze tracking error. Note that all our improvements are orthogonal and complementary to improvements in either the model architecture (e.g. SAGE [22]) or CNN model backbone (e.g. EfficientNet [59]) used; thus, with future improvements in individual components, the accuracy can be further improved.

[1] *iMon*'s source code and a video of it operating as part of a real application is available at https://github.com/imonimwut/imon.

Specifically, our improvements were driven by the following observations. First, we observed that current solutions are hampered by the fixational eye movements present when an individual is focusing on a particular point as a single point is used to represent a person's current gaze. These eye movements are called *microsaccades* and they can impose an average magnitude of 0.6° when a person is observing a scene under normal conditions [46]. The eye movements at this magnitude result in a large 0.42 cm shift in the true position of a person's gaze compared to the target positions (ground truth) when the distance between the user's eye and the screen is ∼40cm (e.g., front-facing camera of a mobile device).

To overcome this, during training, instead of using a 2D coordinate similar to prior work where a person's gaze can only be at a single point at any given time, we use a 2D probabilistic heatmap representation for gaze labels. The 2D heatmap is constructed by a Gaussian Density Function that represents the gaze focus as a region with higher probability and thus accounts for the uncertainty in the exact gaze point position. Contextually, the use of such a heatmap representation allows the gaze estimation model to account for ground truth errors that existing gaze tracking datasets embed. While heatmap representations of the human gaze have been used in previous work to summarize gaze data, to the best of our knowledge, this is the first work to apply such representations to appearance-based gaze estimation models. We show using the GazeCapture [32] dataset, that this technique alone improves the gaze tracking accuracy by ∼7.81%.

Second, we observed that gaze tracking is a pipelined operation (as shown in Figure 1) where input frames are first processed using face detection and facial landmark alignment schemes before they are sent to a deep learning-based gaze estimation model that outputs the final results. We show that improving solely the face detection and landmark alignment steps in the state-of-the-art approaches can improve the overall estimation accuracy by ∼6.16% compared to previous pre-processing methods. *iMon* also employs an image enhancement model based on the UNet architecture [51] that pre-processes eye-region images to improve the visual details and remove motion blur. This step is particularly important as eye-region images captured by a mobile device's front camera tend to have limited resolution and are frequently subjected to motion blur. Our evaluation shows that this image enhancement step improves the overall gaze tracking accuracy by an additional ∼6.38%. These pipeline improvements are used during both training and inference time.

An additional benefit of improving the pipeline as a whole is that we can apply optical flow-based gaze position tracking, during inference time, to keep track of the facial landmarks over consecutive frames. Our evaluations show that this can eliminate 50% to 80% of the pre-processing computational overhead with a small accuracy loss ranging from ∼1.93% to ∼13.23%. Such operations can be selectively enabled depending on whether the highest accuracy or lowest latency is required at the target application.

Finally, we note that the error caused by individual differences in visual focus known as the *Kappa angle* cannot be addressed via a general gaze tracking model. We therefore propose a simple calibration scheme that considers the screen space as a grid and simply adjusts the gaze shift caused by the Kappa angle on the horizontal and vertical axes on each grid cell. When calibrated using 20 fixation points (this takes a few seconds per point for a user to do), *iMon* can further reduce the fixation gaze tracking error by ∼12.4%.

Overall, as aforementioned, our study shows that improving individual components in the gaze estimation pipeline, can bring a noticeable positive impact on the gaze estimation accuracy. Unfortunately, previous work in appearance-based gaze tracking fails to acknowledge the comprehensiveness of the pipeline while mostly focusing on only the performance (and the improvement) of the deep learning model. We take the experiences from previous work, combine them with novel and well-known state-of-the-art solutions to show that a comprehensive approach in examining the entire pipeline is essential in practically achieving high gaze estimation accuracy.

We evaluate *iMon* using the publicly available GazeCapture dataset [32] that was collected from nearly 1,500 smartphone and tablet users. Overall, *iMon* achieves a person-independent (i.e., no Kappa angle calibration) per-frame gaze tracking error of 1.49cm and 1.94cm on the smartphones and tablets, respectively, which is ∼22% better compared to previously reported state-of-the-art results as shown in Table 1. When averaging multiple per-frame predictions of each fixation point, the per-fixation gaze tracking error of *iMon* reduces to 1.26 cm and 1.70 cm and can be further reduced to just 1.11 cm and 1.59 cm by applying a simple personal calibration step. Furthermore, we also exploit a second dataset TabletGaze [24] to validate *iMon*'s performance and observe similar improvements compared to prior work.

Finally, we implemented *iMon* on iOS using Apple's CoreML framework [27] and show that *iMon* can run end-to-end gaze tracking locally on a mobile device at 12 frames per second (fps). Using more latency-friendly settings by applying optical flow to detect eye regions and using MobileNetV2 for the gaze estimation model, *iMon* can perform at nearly 60 fps. In particular, we show that *iMon* performs in real-time completely *locally*. This is important as needing a cloud service introduces additional costs and privacy considerations – especially since facial images are processed. Lastly, we also show through an application-focused user study that *iMon* enables the controlling of a mobile game application with only the users' gaze information.

The key contributions of this work are as follows:

- We present appearance-based gaze tracking as a pipeline of inter-connected components and identify three main sources of error within the gaze estimation pipeline that have critical impact on the performance of appearance-based gaze tracking generally and for mobile platforms in particular.
- We present the design of *iMon*, an end-to-end real-time gaze tracking system for mobile devices, that consists of a mixture of novel and well-established techniques to address each error source and improve the performance of the tracking pipeline as a whole: i) 2D heatmap-based gaze representation, ii) improved image pre-processing (face detection, facial landmark alignment, optical flow tracking, and eye-region image enhancement), and iii) per-user Kappa angle calibration.
- We show, via extensive evaluations with two public datasets, GazeCapture and TabletGaze, that *iMon* outperforms state-of-the-art approaches by up to 22% in accuracy. When calibrated using 20 random fixation points, *iMon* can further reduce the fixation gaze tracking error by ~12.4% to just 1.11cm on smartphones and 1.59cm on tablets. In addition, using real implementations and a user study, we show that *iMon* can run on mobile phones while providing high enough accuracy to enable gaze controlled mobile apps.

2 RELATED WORK

2.1 Gaze Tracking

Gaze tracking is the task of (continuously) estimating the gaze direction represented as a 3D gaze vector [65, 67] or the gaze point on a screen represented as a 2D coordinate [24, 32]. While the 3D direction metric suits with the settings in which the distance between user's eye and the camera is static, this assumption is violated in the mobile context. Due to the dynamic relative positions between the mobile device's screen and user's eye, gaze direction cannot be translated directly to a 2D gaze point without using a separate designated mapping model. As our goal is to develop a practical end-to-end gaze tracking system for mobile applications, our literature review focuses on 2D coordinate gaze estimations.

Largely, approaches for estimating gaze can be categorized as either geometry-based or appearance-based schemes [19].

• **Geometry-based Gaze Tracking.** Geometry-based gaze tracking schemes estimate the gaze direction or gaze point by tracking a certain set of eye features based on a geometric eye model. Such schemes typically require special hardware such as a camera and illumination sources to project and capture the infrared or near-infrared light pattern on the eyes [17, 44]. The reflection image is then used to reconstruct the eye geometry model and estimate eye-related parameters (e.g., iris [60, 61], sclera [50], pupil [41]). However, geometry-based gaze tracking is generally limited for real-world usage due to the dependency on external sensors and light sources for capturing high-quality eye images.

• **Appearance-based Gaze Tracking.** Appearance-based gaze estimation [38, 58, 65, 66] exploits regression models to map a high dimensional input vector to a gaze direction or a gaze point coordinate. The input vector here is either a set of features extracted from eye-region images or the full facial images themselves. Appearance-based gaze tracking requires much more training data compared to geometry-based approaches, but it can work under different lighting conditions and across a broad range of users and facial features. On mobile devices, appearance-based gaze tracking schemes can leverage eye images captured by the front camera while the mobile device is in use without requiring any external sensor support.

2.2 Appearance-based Gaze Tracking Using CNNs

Recent advancements in deep neural network research, especially, convolutional neural networks (CNNs), and the availability of large datasets has enabled significant progress in appearance-based gaze estimation [32, 65, 66]. Datasets such as GazeCapture [32] and MPIIGaze [67] have become representative benchmark datasets for in-the-wild 2D gaze point and gaze direction estimation, respectively. Specifically, GazeCapture, introduced by Krafka et al. [32], is a large-scale 2D gaze dataset with ~2.5 M images collected from nearly 1,500 subjects using Phones and iPads. The work also proposed a 2D gaze estimation model using a CNN architecture, called iTracker, which processes a set of four images extracted from each frame: a face image, a grid image presenting the face position within the frame, left and right eye images. The authors showed that iTracker CNN-based model can learn to map eye images directly to the 2D gaze coordinate and outperform classical models exploiting handcrafted features.

Subsequently, many improvements were made to the 2D gaze estimation accuracy by applying larger image input sizes [29], different model architectures [22], more robust CNN modules [18], or different training schemes [18]. These prior work focus mainly on designing an estimation model suitable for general use that do not require personal calibration, which is challenging given the anatomical differences on a per-user basis. Furthermore, as we later show, these schemes exploit ideal representations of the human gaze for ground truth labeling, which does not represent real micro gaze movements. At the moment, it is claimed that the accuracy of person-independent gaze estimation is still insufficient for use in practical mobile device scenarios [35, 64]. Table 1 summarizes the reported results of different previously proposed methods evaluated using the GazeCapture dataset. Specifically, the previous best person-independent 2D gaze estimation error was 1.77 cm on smartphones and 2.66 cm on tablets as reported by Gou et al. [18]. Lastly, we note that previous work on mobile 2D gaze tracking has mostly focused on gaze estimation model improvements, with less consideration on the organic interaction between many software components within the gaze tracking pipeline that is required for accurate gaze estimations. This is also important from the latency and energy consumption aspects as running the whole gaze tracking pipeline with multiple computationally intensive components in real-time continuously on mobile devices is challenging.

Our work takes a more holistic approach in achieving accurate and efficient gaze estimation on mobile platforms by comprehensively improving the end-to-end gaze estimation pipeline, rather than focusing only on the estimation model which is only a single component in the pipeline. Specifically, we i) re-consider how human gaze is represented in 2D, ii) enhance eye region image quality given mobile usage characteristics, and iii) exploit anatomical eye features for per-user calibration.

3 ERROR SOURCES THAT IMPACT GAZE TRACKING ACCURACY

In this section, we will identify and discuss common sources of errors that impact the accuracy of appearance-based gaze tracking. Then we will show in Section 4 how *iMon* overcomes these errors and present evaluation results in Section 5.

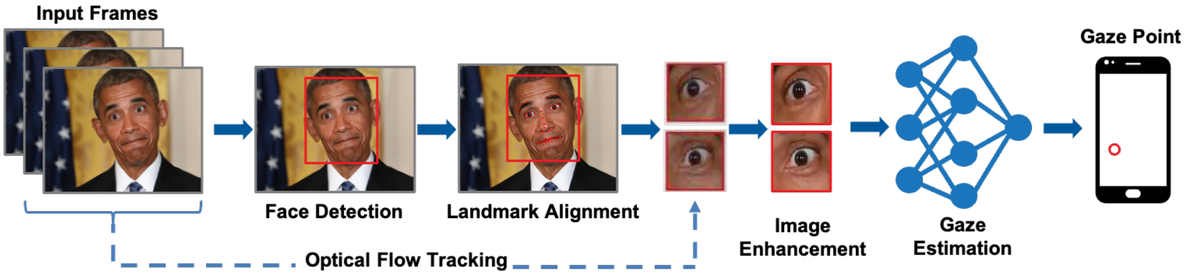


Fig. 1. Appearance-based gaze tracking pipeline on mobile platforms: each frame extracted from the video captured by the mobile device's front camera is fed into a face detection model to locate the face position within the frame. Then a facial landmark alignment model is applied to align the position of the landmarks within the face bounding box. Finally, a gaze estimation model takes the eye regions as input and outputs an estimation of the user's gaze point. Potential optimizations can be made, as in this work, by identifying image similarity and skipping through redundant operations using techniques such as optical flow.

3.1 Error Source 1: Errors in Labeled Data Used for Training

Developing an accurate gaze estimation model requires a large and diverse training dataset to capture all possible facial feature combinations. However, collecting high-fidelity labeled gaze data (i.e., images with ground truth locations of the gaze) is not easy and is usually done in two ways: **(1)** using specialized gaze tracking devices (e.g., Tobii X2-60 [26], EyeLink Portable Duo [15]), or **(2)** asking participants to actively focus their gaze on a target point on the screen as part of a data collection experiment.

Given the high costs of purchasing and calibrating gaze tracking platforms, it is challenging to collect samples from a large and diverse population using the first approach. On the other hand, the second approach can be made scalable by exploiting mobile applications that present participants with a sequence of target gaze fixation points on the screen with recordings from the front facing camera. Thus, several recent work has used this approach [24, 32] to train a robust gaze estimation model using a large set of diverse participants.

Unfortunately, this participatory data collection approach, while scalable, is susceptible to errors caused by microsaccade movements. These are the small and involuntary eye movements that occur while the eye is focusing on a fixed point [40]. Therefore, even if the user believes to be observing a single point on the screen, the gaze shows small movements, which complicate the ground truth. Prior work has shown that such microsaccades contribute to enhancing the spatial detail of our vision, and prevent it from fading [40, 52]. Under stationary fixation scenarios, microsaccades have an average amplitude of 0.61° [46]. This translates to a shift in focus, from the point being fixated on, of up to 0.43 cm when the screen is positioned 40 cm away from the human eye. Note that the average peak velocity of microsaccade movements ranges from 40° to more than 200° per second, and the frequency of such movements can vary from 1 to 3 times per second depending on the visual task [46].

Such rapid movement characteristics make obtaining fine-grained labeled data very challenging. Thus, in practice, when collecting labeled data for appearance-based gaze tracking, a common assumption made is that the participants focus (and stay focused) on the target point *as instructed*. However, due to microsaccade movements, there are inevitable deviations from the (believed) ground-truth point. We make our first observation of an error source as follows:

Error Source #1: *The current commonly used gaze label data collection methods cannot capture the exact ground truth position of the target gaze point.*

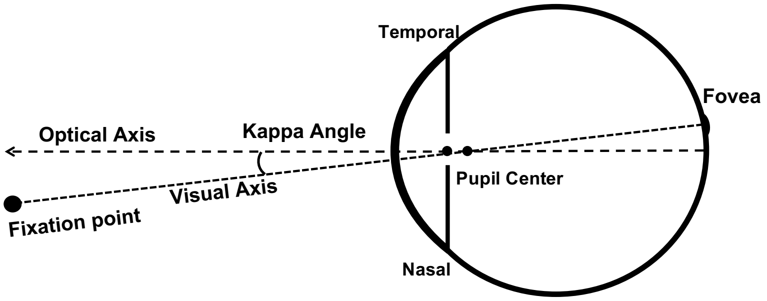


Fig. 2. Illustration of the eye's Kappa angle.

3.2 Error Source 2: Errors in the Input Eye Image Data

As Figure 1 shows, a typical processing pipeline used by an appearance-based gaze tracking system commonly consists of multiple interconnected components. Given an input frame, the face region on the image is initially identified and key facial landmarks are aligned. Next, the landmarks related to the eye, as well as the eye-region images themselves, are fed into a gaze estimation model, which usually uses CNNs [22, 32] to output the 2D position estimate of the user's focus.

Modern smartphones and tablets have front facing cameras with high pixel resolutions. However, even with this, the size of the eye regions is very small when compared to the full image size. For example, in the widely used GazeCapture dataset [32], more than half of the images (with 640x480 resolution) captured by the front camera of iPhones and iPads have eye regions with a resolution of lower than 36x36 pixels. This limits the visual content that can be extracted from the eye images, which negatively affects the performance of a gaze estimation model. Furthermore, images captured on mobile devices often contain motion blur caused by hand and head movements. This blur makes the detection of already-small eye regions even harder. With this in mind, we make our second error observation as follows:

Error Source #2: *The eye regions captured by a mobile devices' front camera will usually have low resolution and exhibit motion blur effects.*

In addition, face detection and facial landmark alignment models also introduce high computational overheads as they execute complex CNN operations on a per-frame basis. Thus, a gaze tracking pipeline that processes each frame through all pipeline operations could result in high latency and a computationally expensive solution that would be difficult to run in real-time on resource-constrained mobile devices due to frequent use of a deep learning model [25, 49]. We thus note the following optimization observation:

Optimization Observation: *Running the full gaze tracking pipeline on every frame could impose significant computational and latency costs on a mobile device. Schemes to suppress the computation with minimal loss in accuracy can allow for effective real-time gaze tracking.*

3.3 Error Source 3: Errors Caused by Individual Eye Variations

Figure 2 illustrates the Kappa angle, which is the angular difference between the optical and visual axes at the eye (i.e., the difference between where your eyes appear to look at and what you are actually seeing). The optical axis, also referred as the pupillary axis, is the line perpendicular to the cornea that intersects the center of the entrance pupil [43]. The visual axis is the line connecting the point of gaze and the fovea, passing through the center of corneal curvature (nodal point) of the eye.

The Kappa angle varies across people and a study by Gharae et al. of 977 participants, suggests that the average and standard deviations for the Kappa angle was -0.02° ± 0.49° on the horizontal axis (x-axis), and -0.09° ± 0.32° on the vertical axis (y-axis) [16].

Unfortunately, determining the Kappa angle from the input images is not easy as these eye images can only be used to determine the optical axis, but not the visual axis. Hence, gaze estimation models can only learn to approximate the Kappa angle from the training dataset. From the standard deviation figures produced by Gharae et. al [16] mentioned above, the average estimation error is 0.34 cm and 0.22 cm on the horizontal and vertical axis respectively, assuming the eyes and screen are 40 cm apart and the optical axis is perpendicular to the screen. Based on this, we make our third and final error observation as follows:

Error Source #3: *Improving the accuracy of gaze estimation will require accounting for the error caused by individual differences in Kappa angle.*

4 *IMON*

Based on the aforementioned observations, in this section, we present the design of *iMon* and show how it overcomes the errors described in Section 3. We present *iMon* as a gaze estimation system that exploits any available gaze estimation model (as part of the gaze estimation pipeline) and improves the estimation accuracy using a combination of techniques within the pipeline as we discuss in this section.

4.1 Using Probabilistic Heatmaps to Overcome Microsaccade Errors

As stated in Section 3.1, the ground truth data even for popular eye image datasets have errors due to users' microsaccade movements. These involuntary fixational movements suggest that the actual visual focus of the human eye should not be defined as a single point, but should instead be a focus *region* of all the points observed by the eye when focusing on a target point on the screen. A single 2D coordinate, as used in existing datasets, can only (at the best) represent the center of a person's visual focus and the actual focus will move around this central point due to microsaccade movements. With this observation, we posit the following hypothesis to address Error Source #1 described in Section 3.1:

Hypothesis #1: *A probabilistic representation of the human gaze is needed (instead of the currently applied 2D coordinates) to correct the errors caused by microsaccade movements in existing labeled gaze data. This correction will result in improved gaze tracking accuracy.*

To represent human gaze in a more realistic and probabilistic form, *iMon* exploits a 2D probability distribution heatmap to represent the gaze focus region instead of using a single 2D point. Given that the microsaccade movements of users can vary, we generalize the 2D gaze representation by constructing the 2D Gaussian function with the mean values of the two axes (x_0, y_0) corresponding to the coordinate of gaze point label, and the standard deviation (σ_X, σ_Y), which represents the magnitude of microsaccade movements. We take the average degree of human microsaccade motion reported from previous work in ophthalmology [46] and apply an offset with respect to the distance between the user and screen, since changes in distance will lead to varying (σ_X, σ_Y) given the same physical microsaccade motion.

Figure 3 presents an example of the focal heatmap of the entire screen (left) and the close-up focus region (right) showing the impact of microsaccade motions. Specifically, to apply such heatmap representations to gaze labels for model training (instead of the conventionally used 2D coordinates), we use a per-pixel loss function as shown below. We compute the loss value L of the estimated heatmap $\hat{h}$ (gaze estimation output) as compared to the labeled heatmap h via the sum of all absolute pixel errors. Note that the sum of all weights (pixel values) on each labeled heatmap is 1.

$$L(h, \hat{h}) = \frac{1}{2}\Sigma_i|h_i - \hat{h}_i| \text{ , } where\ h_i\ denotes\ the\ value\ of\ pixel\ i\ in\ heatmap\ h \tag{1}$$

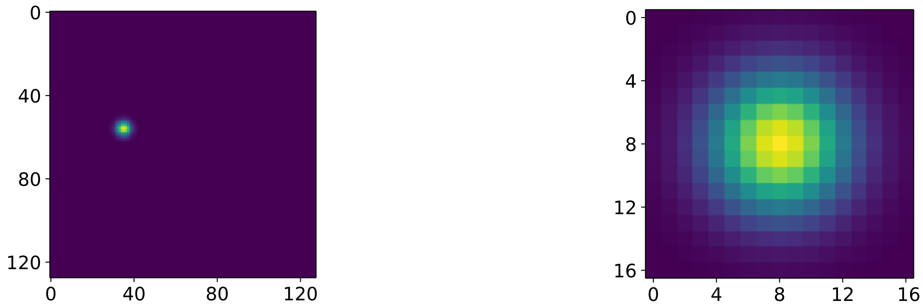


Fig. 3. 2D heatmap gaze representation of an entire screen (left) and its zoomed-in gaze focus (right). Each pixel denotes 0.2 cm on the screen.

The default heatmap's gaze focus area we use in this work is sized for a person 40 cm away from the screen. If the person is nearer or farther, we scale the focus area accordingly. As there is no absolute measurement of the distance between the front camera and user's face available, we approximate this distance based on the size of the face region captured in the frames. We note that the effect of individual face size differences is far less significant as compared to the impact of the face-camera distance between the camera and user. For example, the size of the face region would double if a user moves the device closer – from 40cm to 20cm away from his/her face, which is not commonly seen for different face sizes at similar distances. We also point out that this approximation relies on the assumption that the zooming features of the front camera is consistent for the training and inference data.

One positive side effect of this method is that the loss function will output a heatmap with all pixels having zero values if the input image is invalid (e.g., blurry images or when a user is not looking at the phone). This is possible given that the gaze focus takes only a small region on the entire heatmap (screen) and all pixels outside of the focus region are labeled with zero (Figure 3). With a large dataset consisting of gaze focus regions uniformly distributed over the entire device screen, the model will confidently output an all-zero heatmap for invalid images where the gaze cannot be properly detected. Such invalid heatmaps can be leveraged in practice to improve the pipeline efficiency by detecting invalid inputs and skip unnecessary pre-processing steps such as blink and blur detection.

Note that a predicted heatmap can be converted back to a 2D coordinate by simply computing the weighted center of the heatmap. We apply this post-processing step to evaluate the gaze tracking accuracy by computing the Euclidean error when comparing with previous work. We validate this hypothesis and show that exploiting 2D heatmaps over coordinates improves the overall accuracy by ~7.81% improvement in Section 5.4.

We also note that, while the idea of using 2D heatmaps to visualize gaze data is not new [4, 32, 55], this work is the first to incorporate such gaze representations into the training and inference processes of appearance-based gaze estimation and observe its impact.

4.2 Improving the Pipeline and Input Images

Appearance-based gaze tracking models rely heavily on the visual characteristics embedded within the input eye images. However, in practical use cases, eye images captured through a mobile device's front facing camera frequently suffer from issues such as low resolution or motion blur due to either the movement of the camera or the person. Our hypothesis for addressing this error, also discussed as the second error source in Section 3.2 is as follows:

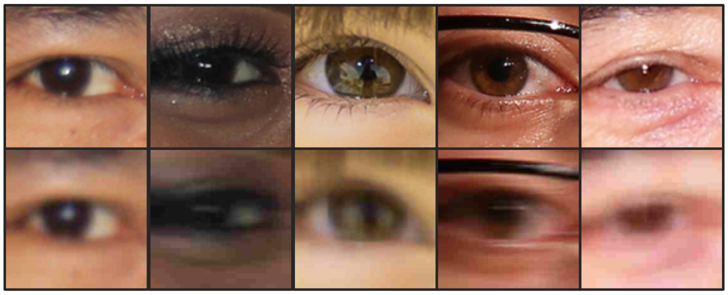


Fig. 4. Examples of high-quality (top) and *crappified* images (bottom).

Hypothesis #2: *Adding components to the gaze estimation piepline for enhancing the visual details of low-quality eye images, that commonly occur in mobile usage scenarios, can significantly improve the gaze tracking accuracy.*

To assess this hypothesis, *iMon* exploits an image enhancement model that converts a given eye-region image (that exhibits various levels of low resolution and motion blur effects), to a corresponding image with better visual quality and refined details. These enhanced images are then fed to the gaze estimation model for evaluation. We integrate this process into *iMon*'s gaze estimation pipeline.

The image enhancement model in *iMon* is based on the UNet architecture [51], which is known to be effective in learning nonlinear relationships between low- and high-resolution images [2]. The model composes of an encoder that uses convolutional blocks and a symmetric decoder using deconvolutional blocks (also referred as transposed convolutional blocks). We maintain a small residual network, ResNet18 [23], as the backbone model for the encoder and decoder. Furthermore, skip connections are added between the convolutional and the corresponding symmetrical deconvolutional blocks.

To train this image enhancement model, we apply a semi-supervised learning approach called the *decrappify* method [1]. Given an original high-quality image, the idea is to create a low-quality version of the same image by applying *removal* effects. The two target scenario effects we apply are “coarse visual details” reflected by the use of low resolutions, and “motion blur” caused by unintended user movements. To generate a low-quality version (with the same resolution) of the original high-quality image, we first downscale the high-quality images to a smaller size and then upscale them back to the original size using bicubic interpolation. Second, the motion blur effect is added by convolving the original image with horizontal and vertical motion blur matrix kernels. The kernel sizes are randomized to introduce various blur intensities. Figure 4 shows examples of this process, where on the top we present the original high-quality images and on the bottom the corresponding crappified images are presented.

iMon's image enhancement model is trainable end-to-end, and convergence is achieved by minimizing the combination of i) pixel loss and ii) perceptual feature loss between the reconstructed image and the original image with high quality. The perceptual feature loss is computed using a VGG16 model [56] pre-trained with the ImageNet dataset [13]. The objective of using this combination of loss functions is to encourage the model to learn to reconstruct an image of higher quality, so that it has knowledge of similar pixel values relative to the ground truth image. More importantly, when the reconstructed image is fed through the pre-trained VGG16 model, it produces a similar feature representation to the ground truth. Thus, the input to the image enhancement model is a blurry image with coarse details (directly taken from the smartphone camera), and the outcome of the final deconvolutional layer is a refined image with better visual quality. We quantify the performance improvement that this model brings in Section 5.5.

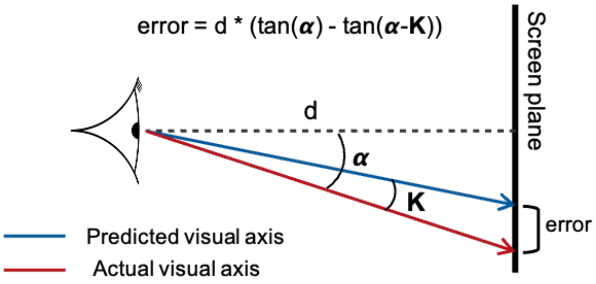


Fig. 5. Individual difference in vertical kappa angle results in gaze estimation error.

In addition to this novel image enhancement model, *iMon* further improves the gaze estimation pipeline by exploiting state-of-the-art face detection and facial landmark alignment algorithms. Specifically, we use the Single Shot Scale-invariant face detector (S3fd) [63] to detect the face bounding box, and then apply the Face Alignment Network (FAN) [7] to specify eye regions. We selected these algorithms as they have demonstrated robust performance across multiple large-scale 2D face detection and facial landmark alignment benchmark datasets [28, 53, 62].

4.3 Improving Accuracy through Personal Calibration

As mentioned in Section 3.3, the Kappa angle, an important factor that determines the actual gaze coordinate, differs for various users. Unfortunately, per-user Kappa angle magnitudes cannot be determined solely from observing images of the eye as the visual axis is hidden. Therefore, regardless of the size of training data, a general appearance-based gaze estimation model, even at its best, can only map the optical axis to the visual axis with the averaged angle difference learnt from the training data. Thus, users with Kappa angle magnitudes deviating farther from the average will experience a higher gaze estimation error.

Recall that the standard deviation of the vertical and horizontal Kappa angles are 0.32° and 0.49°, respectively, representing the average angle between the visual axis predicted by an ideal model and the actual visual axis of the user. In an ideal scenario where the user's eye's visual axis is perpendicular to the screen, the error caused by the individual kappa angle is minimal, approximately 0.22 cm and 0.34 cm for vertical and horizontal, respectively, or a 0.4 cm euclidean distance error. In other cases where the visual axis deviates significantly (e.g., more than 20°) from the orthogonal line of the screen plane, either horizontally or vertically as illustrated in Figure 5, this error can be much higher (given the same Kappa angle). Specifically, the more the visual axis deviates from the orthogonal line of the screen, the higher the error caused by individual difference of Kappa angles. Given the large diversity of mobile phone users, this error cannot be ignored when designing a practical solution.

To address this issue, *iMon* adopts a calibration scheme where the screen area is split into three rows and three columns (a 3×3 grid). Each row and column processes a different calibration value for the vertical axis and the horizontal axis respectively. Therefore, each cell of the 3x3 grid will have a unique pair of calibration values on the two axes. We assume that the error caused by individual Kappa angle differences would be similar for every point included in the cell and can be corrected by linearly shifting the predicted gaze point or focus region by measurements taken from a small number of calibration points. Thus, we formulate our hypothesis on overcoming Kappa errors by providing per-user personalization as follows:

Hypothesis #3: *The effect of per-user Kappa angle variations can be addressed using a simple calibration scheme that pre-captures a small number of samples that can be used to estimate the error caused by individual Kappa angle and correct other gaze predictions.*

```
while running_facial_landmark_tracking do
    frame = get_next_frame ()
    facial_landmarks = optical_flow_track (frame, prev_frame, prev_facial_landmarks)
    nme = normalized_mean_error (facial_landmarks, prev_facial_landmarks)
    if nme > threshold then
        facial_landmarks = get_facial_landmarks (frame)
    prev_frame = frame
    prev_facial_landmarks = facial_landmarks
```

Algorithm 1: Continuous facial landmark tracking.

To perform the calibration, we take samples from n ($i.e.$, $n < 20$) calibration points. Given that this calibration process can be considered as overhead, we make sure that this is a one-time process (per-user) and is quick to complete. We choose the calibration points to provide sufficient coverage of the entire screen. We present the accuracy improvements that our calibration scheme brings in Section 5.6.

4.4 Improving Latency Using Optical Flow

Finally, one important observation made in Section 3.2 when we analyzed the data pipeline was that running the entire pipeline on every image is computationally expensive. This is particularly important for mobile device scenarios where the computational resources available are limited and power-constrained. In this section, we present a solution, using optical flow [39], that allows *iMon* to significantly reduce the overhead of performing pre-processing steps (e.g., face detection and facial landmark alignment) with just a small loss in accuracy.

Our insight is to exploit the fact that with a high enough input frame rate (e.g., 15-30 fps) from the camera, the differences in motion between consecutive frames, will be marginal in many cases. This is especially true for mobile usage scenarios. For example, consider a scenario where the user is browsing an online shopping app and the front camera is used to analyze the user's gaze. The posture of the person will seldom change while they are scrolling through the items, and their eye landmark positions will show minimal changes between frames. These mobile device usage patterns provide us with an opportunity to short circuit unnecessary pipeline operations if the change in motion is not significant.

To do this, *iMon* uses the Lucas-Kanade optical flow algorithm [39] to keep track of only the facial landmarks (12 points for the left and right eyes) over consecutive frames. The robust and relatively computationally heavy face detection and landmark alignment models only run when the optical flow suggests a significant change in facial landmarks.

This optical flow-based optimization is presented in Algorithm 1. As the algorithm shows, *iMon* tracks the facial landmarks in the current frame using information from the previous frame. As long as the distance between the two sets of landmarks is under a preset threshold, *iMon* reuses the previous facial landmarks and skips the face detection and facial landmark alignment model operation for this current frame.

This optimization greatly improves the overall gaze estimation pipeline latency at the loss of a small amount of accuracy. We show the effectiveness of this optimization in Sections 5.7 and 5.8.

5 EVALUATIONS

In this section, we present a full evaluation of *iMon* using two publicly available large-scale gaze datasets and an application-focused user study to demonstrate the effectiveness of our proposed solutions at satisfying our hypothesis. We first examine the performance of *iMon* in various configurations and dimensions using the public datasets, and later present the user study in Section 5.9.

5.1 Evaluation Setup

• **Public Datasets.** The main dataset we use for our evaluation is the GazeCapture dataset [32], which provides gaze data collected from 1,474 iPhone and iPad users through an Amazon Mechanical Turk study. It contains nearly 2.5 M 640x480 resolution images of subjects focusing on ~200K unique fixation points, with multiple (continuous) images available for each fixation point. To induce practical variations, subjects were asked to continuously change their head and device orientation, and change their relative distance from the screen during the data collection phase. For all our evaluations, we use the same the train/validate/test data split originally provided in GazeCapture, where the subjects are split into 1,271 training, 50 validate, and 150 test subjects respectively.

Additionally, we perform cross-dataset evaluation using models pre-trained on the GazeCapture dataset and then evaluated on a secondary dataset – the TabletGaze [24] dataset. The TabletGaze dataset was collected using a Samsung Galaxy Tab S and contains data from 51 subjects consisting of ~100K images, 35 fixed gaze points, ~20K unique point-of-gaze or fixation points (35 fixed points × 51 subjects × 4 body postures (standing, sitting, slouching, and lying) × 4 sessions - with some invalid data). This secondary evaluation was performed to validate our findings from the first dataset.

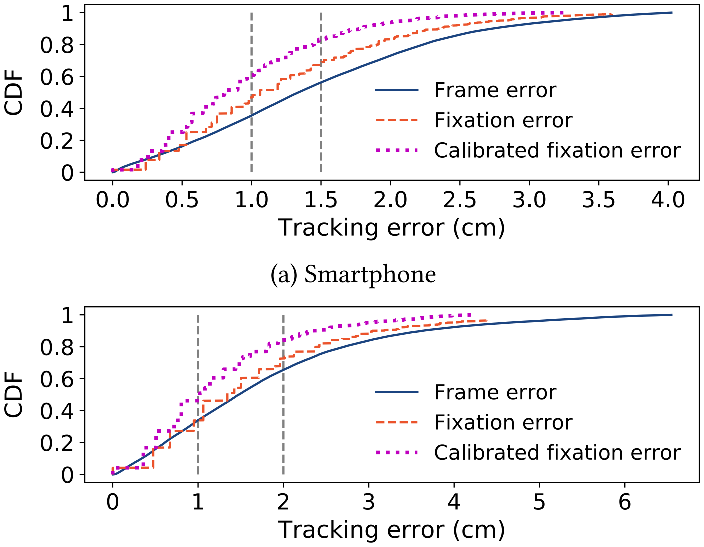
• **Evaluation metrics.** To evaluate the accuracy of *iMon* and different gaze estimation models used for comparison, we report two types of error values: i) frame error and ii) fixation error. The frame error is simply the Euclidean distance between the ground truth gaze position and the one estimated by the model on a per-frame basis. The fixation error (also referred as dot error in Krafka et al. [32]) is computed by averaging multiple frame errors that correspond to the same fixation point. Thus, we can think of the fixation error as a practical metric for gaze tracking applications where input frames are captured continuously. Even for a very short fixation duration (e.g., 300 ms), multiple gaze estimations can be made and aggregated to achieve a more accurate result.

• **Model training.** We trained all the models used for evaluation from scratch using 100K iterations and a batch size of 64. We use the Adam optimizer [31] with an initial learning rate of 0.001 and reduce it by 10% for every 4K iterations. We also apply the mixed-precision scheme for the models so that only the final activation layer is a 32-bit floating-point type with the other layers represented as 16-bit floating-point types. The size of the input was 112×112 pixels. Note that we do not apply any data augmentation or per-device/per-screen-orientation model fine-tuning.

5.2 *iMon*'s Overall Performance

Table 1 (presented in Sec. 2) summarizes the overall gaze tracking performance of *iMon* using the GazeCapture dataset, compared with prior state-of-the-art mobile device appearance-based gaze tracking solutions. Specifically, *iMon* achieves a person-independent frame error of 1.49cm and 1.94cm on smartphones and tablets respectively, which outperforms the previous state-of-the-art by 22.29% on average.

Note that the work proposing the GazeCapture dataset, Krafka et al. [32] in Table 1, by default, identifies the facial and eye region landmarks from each frame using Apple's built-in face detection algorithm and performs facial landmark alignment using a model proposed by Baltrusaitis et al. [3]. As mentioned in Section 4.2, we noticed that improvements to these preprocessing algorithms can have a significant impact on the overall accuracy, and applied the S3fd face detection model [63] and the FAN facial landmark alignment model [7] instead of the originally used algorithms. From their output, we identify and remove 263,858 frames with blinking activities. As a result, *iMon* exploits a total of 2,077,941 effective frames (1,724,239 for iPhones, 353,702 for iPads) compared to 1,490,959 frames (1,237,171 for iPhones, 253,788 for iPads) when the default algorithms are applied. We observed that this simple improvement in applying state-of-the-art algorithms leads to 6.2% increase in accuracy by itself, which is integrated in *iMon*'s final accuracy results.



(a) Smartphone

(b) Tablet

Fig. 6. CDF of *iMon*'s tracking error evaluated on GazeCapture dataset.

When averaging multiple per-frame gaze predictions that belong to the same fixation point, *iMon* achieves a fixation error of 1.26cm and 1.70cm. We later show that the fixation error can be further reduced to 1.11cm and 1.59cm when applying personal Kappa angle calibration with 20 calibrating points (see Sec. 5.6).

Figure 6 shows the distribution of three types of errors when evaluating *iMon* on the GazeCapture dataset. We note that around 60% of the samples collected from smartphones and half of the samples collected from tablets have calibrated fixation errors of less than 1cm.

One interesting obseration we make observe that the gaze tracking error on tablets is consistently higher than that of smartphones. One main reason for such behavior is that the amount of data collected from tablets in GazeCapture dataset is much smaller compared to data from smartphone (only 15% of the total data are from tablets, while the other 86% are from smartphones). Another reason is due to the bigger size of tablet screen, hence, the prediction value range tends to show larger variations.

Additionally, we performed a 5-fold cross-subject evaluation on the TabletGaze dataset. In particular, we randomly split the dataset into five subject groups. We use the data of four groups to fine-tune the gaze estimation model pre-trained using the GazeCapture dataset, then evaluate the model with the data from the remaining group. The process is repeated for each fold. This fine-tuning operation is important because the TabletGaze dataset was collected using tablet devices with different display dimensions and camera positions which are not included in GazeCapture dataset. A similar fine-tuning step was also applied in previous work (i.e., Krafka et al. [32]); thus, our evaluations provide a fair comparison. The results in Table 2, show that *iMon* also achieves significant improvements on the TabletGaze dataset with a 28.36% reduction in tracking error compared to prior work. In the rest of this section, we present evaluation results in the impact of each *iMon* component.

5.3 Selecting the Base Gaze Estimation Model for *iMon*

As mentioned, *iMon* focuses on improving the entire gaze estimation pipeline; thus, is designed to be usable with any appearance-based gaze estimation model. To identify the most accurate and efficient model available for our evaluations, we evaluated the two best performing mobile gaze tracking approaches from recent work – namely the iTracker [32] and SAGE [22] (shown in Figure 7) gaze estimation model architectures – and integrated them

Table 2. Evaluation results (Euclidean error in cm) on TabletGaze dataset. Fixation errors have not been reported for [24] and [32].

Method	Frame Er.	Fixation Er.
Baseline - center of screen	7.15	7.15
TabletGaze [24]	3.17	-
iTracker [32]	2.58	-
iTracker - our implementation	2.72	2.68
iMon	**2.01**	**1.78**

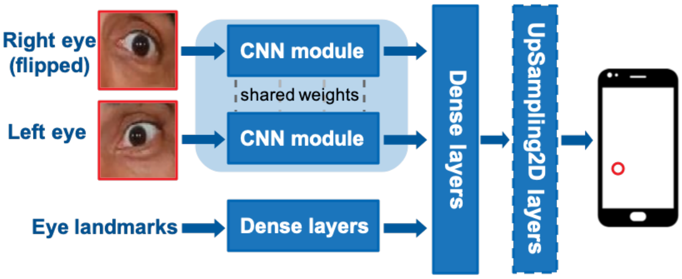


Fig. 7. SAGE architecture with UpSampling2D layers for 2D heatmap gaze representation output.

to the *iMon* pipeline. Note that previous literature suggests that the SAGE architecture, which uses only the eye-region images and the eye landmarks, outperforms the iTracker-based architecture which exploits both eye- and facial-region images [22].

Both the iTracker and SAGE architecture use AlexNet [33] as their CNN backbone. In addition, we also examined the effect of using different backbone CNN models with these architectures for gaze estimation. Specifically, we used EfficientNet-B3 [59] and MobileNetV2 [54] as comparisons as they are known to be efficient and robust.

Our results in Table 3 indicate that the choice of both the model architecture and the backbone CNN module can heavily affect the overall gaze estimation accuracy. Specifically, the results agree with prior work [22] and show that SAGE outperforms iTracker across all CNN backbone choices. In addition, within the same architecture, the EfficientNet-B3 CNN backbone has the best accuracy and outperforms AlexNet by up to 14%.

We thus select the combination of SAGE and EfficientNet-B3 for evaluating *iMon* as this combination showed the best accuracy. However, we also note that MobileNetV2 requires less computational resources than EfficientNet-B3 and only has a small accuracy drop compared to EfficientNet-B3. We show how we leverage MobileNetV2 with Optical Flow to produce a latency optimized version of *iMon* in Section 5.8. While the gaze tracking model is not a contribution of this work, this evaluation process is important given that our goal is to propose an improved to supplement a well-performing gaze tracking model so that higher accuracy can be achieved. Thus, in all subsequent results, *iMon* will be using SAGE with EfficientNet-B3 unless stated otherwise.

5.4 Evaluating Heatmap Gaze Representation

To evaluate the accuracy impact of using 2D heatmaps (Section 4.1) to represent the gaze, we trained the baseline gaze tracking model with 2D heatmap gaze representations extracted using the GazeCapture dataset. To generate heatmap representations, we added UpSampling2D [11] layers after the Dense layers within the SAGE model architecture (c.f., Figure 7) to convert the 1D output of the Dense layers into a 2D heatmap.

Table 3. Evaluation results (Euclidean error in cm) of gaze estimation models with iTracker and SAGE architectures and different CNN model backbones on GazeCapture dataset.

Model Architecture	CNN Model Backbone	Error (phone / tablet)	
		Frame Er.	Fixation Er.
iTracker	AlexNet	2.04 / 3.32	1.62 / 2.82
iTracker	MobileNetV2	1.75 / 2.57	1.56 / 2.33
iTracker	EfficientNetB3	1.80 / 2.59	1.62 / 2.41
SAGE	AlexNet	1.84 / 2.72	1.63 / 2.49
SAGE	MobileNetV2	**1.69** / **2.37**	**1.48** / **2.13**
SAGE	EfficientNetB3	**1.66** / **2.31**	**1.47** / **2.12**

Table 4. Evaluation results on GazeCapture dataset using 2D heatmap gaze representation with different gaze focus radius (SAGE architecture + EfficientNet-B3). We compute using the center point of the heatmap and report the Euclidean error in cm. Adaptive radius ranges from 0.2cm to 0.4cm depending on the face size within each input frame.

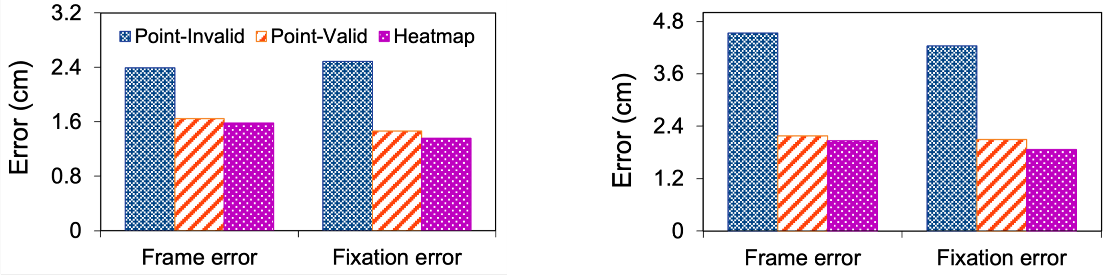
Gaze Rep.	Radius	Error (phone / tablet)	
		Frame Er.	Fixation Er.
Single point	-	1.66 / 2.31	1.47 / 2.12
Heatmap	0.2	1.59 / 2.11	1.41 / 1.90
Heatmap	0.4	1.62 / 2.15	1.41 / 1.94
Heatmap	0.6	1.64 / 2.20	1.45 / 1.91
Heatmap	0.8	1.62 / 2.27	1.42 / 1.98
Heatmap	adaptive	**1.58** / **2.07**	**1.36** / **1.86**

Using the knowledge of the average magnitude of the microsaccade motion (0.61° or 0.43cm shift when the screen is positioned 40 cm away [46]), we investigated the accuracy impact of using fixed heatmap-based focus region sizes ranging from 0.2 cm to 0.8 cm. In addition, we evaluated an additional adaptive case where the size of the focus region was adaptively set depending on the size of the detected face regions in each input frame. The results are presented in Table 4 and show that using a 2D heatmap gaze representation, rather than a single point representation, improves the accuracy (note: top most column shows, as a reference, the results when using single point gaze representations with the SAGE architecture + EfficientNet-B3). Overall, applying adaptive focus regions achieves the highest accuracy of 1.58 cm and 2.07 cm (frame error) on smartphones and tablets. In Table 5 (a), we also compare the performance of the adaptive heatmap method using MobileNetV2 and EfficientNet-B3 backbones, and observe that EfficientNet-B3 still achieves better performance.

As discussed in Section 4.1, one important side benefit of using the heatmap gaze representation is that the gaze estimation model will output a blank heatmap when it encounters an invalid input. We found that even with our initial frame filtering process to remove obviously wrong inputs (discussed in Section 5.2), blank heatmap predictions still occurred for 2.69% of the input frames (1.65% fixation points) in GazeCapture's test dataset. To further validate the performance improvement from this side benefit, we tested the baseline solution using 2D point coordinates (presented in Table 3) with data containing invalid samples (that can be detected only by the heatmap model) and with only valid samples. Figure 8 presents these results and shows that the baseline model has a significantly higher average tracking error of 2.40 cm (smartphones) and 4.54 cm (tablets) when processing data with invalid frames (the Point-Invalid results) compared to when processing data with only valid frames

Table 5. (Top): Evaluation using single point gaze representation for reference; (a): Heatmap effectiveness validation - Evaluation using *heatmap with adaptive radius*; (b): Eye-region image enhancement effectiveness validation. All gaze estimation models are based on SAGE architecture and evaluated on the GazeCapture dataset.

CNN Model Backbone	Error (phone / tablet)		Improv.
	Frame Er.	Fixation Er.	
MobileNetV2	1.69 / 2.37	1.48 / 2.13	-
EfficientNetB3	1.66 / 2.31	1.47 / 2.12	-
(a) With heatmap gaze representation			
MobileNetV2	1.65 / 2.24	1.46 / 1.95	+ 3.11 %
EfficientNetB3	**1.58** / **2.07**	**1.36** / **1.86**	+ 7.81 %
(b) With eye-region image enhancement			
MobileNetV2	1.65 / 2.24	1.45 / 2.06	+ 2.79%
EfficientNetB3	**1.58** / **2.09**	**1.39** / **1.94**	+ 6.38%



(a) Smartphone

(b) Tablet

Fig. 8. Tracking error evaluated on invalid samples (detected via heatmaps) compared to valid samples in GazeCapture test set. Point-Invalid indicates tracking error of baseline gaze estimation model using 2D point coordinate on invalid input samples.

(Point-Valid results). The table also shows the accuracy difference when using 2D heatmaps. This suggests that being able to filter out invalid data in the form of blank heatmaps does provide a positive impact on the overall gaze estimation performance.

5.5 Impact of Eye-region Image Enhancement

To evaluate the impact of the image enhancement component in *iMon*, we trained the UNet model discussed in Section 4.2 with the Flickr-Faces-HQ (FFHQ) dataset [30], which contains 70K high-quality 1024x1024 resolution PNG face images. The model is trained from scratch and converges after 50K iterations with a batch size of 32 using the Adam optimizer. Both the input and output images of the image enhancement model have a size of 112×112.

Figure 9 presents examples of samples from the GazeCapture dataset before (top) and after applying *iMon*'s image enhancement model (bottom). By comparing the top-bottom image pairs, we can visually observe quality improvements for low-quality and blurred images. Note once more that these results are achieved with the enhancement model trained using a single dataset (e.g., FFHQ) with 70K images. We conjecture that this model could potentially improve the fidelity of eye-image input further if it is exposed to more diverse and high-quality eye images through semi-supervised training.

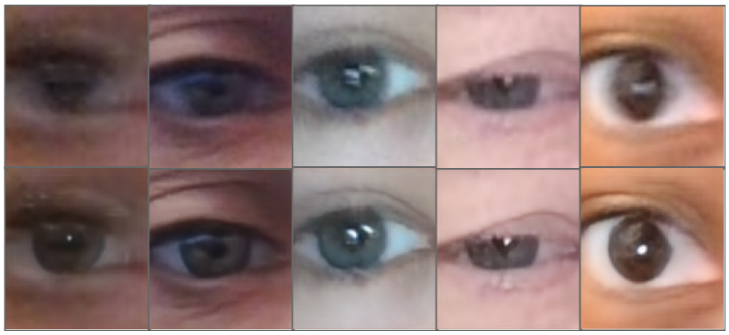
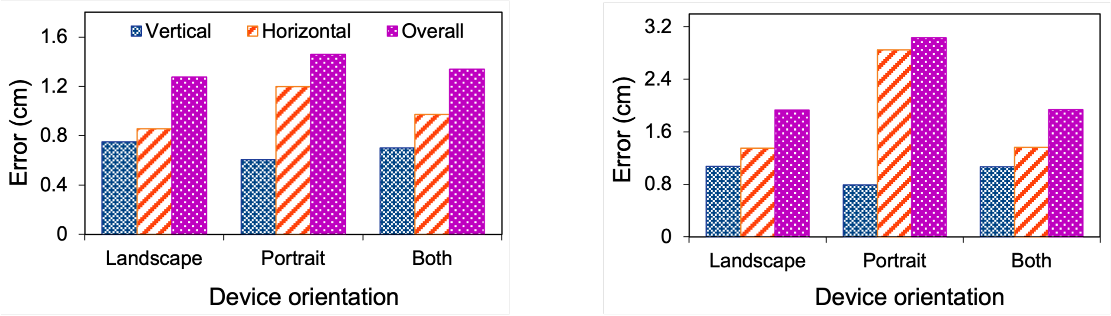


Fig. 9. Examples of eye images from GazeCapture dataset (top) and their corresponding enhanced version (bottom).



(a) Smartphone

(b) Tablet

Fig. 10. *iMon*'s evaluation tracking per-frame error of each device orientation on GazeCapture dataset.

Quantitatively, by evaluating our baseline estimation models with the enhanced images and comparing to the same baseline using original images (c.f., Table 3), we observe, as shown in Table 5 (b), that the accuracy improves by up to 6.38% with image enhancements.

5.6 Impact of Personal Kappa Angle Calibration

To evaluate the impact of calibration (Section 4.3), we first present the gaze estimation error for different screen orientation modes in Figure 10. We observe that the horizontal axis has higher errors than the vertical axis for both portrait and landscape modes. This difference can be partly explained by the Kappa angle error as it is known to show significantly larger amplitudes on the horizontal axis [16].

Therefore, for Kappa angle calibration, we divide the screen into a 3×3 grid in which the gaze points within each cell could be shifted differently due to the effect of the individual differences in Kappa angle. We then have each user focus on specific points on the screen to calculate the Kappa angle error for each portion of the screen. Given our experiments with pre-existing datasets (e.g., GazeCapture dataset) in which we cannot ask users for explicit calibration operations, we implemented this by choosing, a number of fixation points from a user's data located in four specific screen positions (top, bottom, left, and right). We then used these data points to compute the bias (difference between prediction and label) for both the horizontal and vertical axes. This bias is then applied to correct the predictions of other samples for that specific subject, just as we would apply the Kappa angle error for real users.

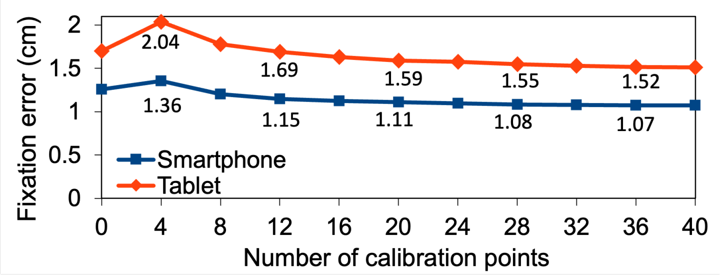


Fig. 11. Fixation error when applying per-user Kappa angle calibration with different number of calibration points on GazeCapture dataset.

Figure 11 shows that *iMon*'s Kappa angle calibration scheme can effectively reduce the gaze estimation error without requiring any model fine-tuning. Specifically, with only 20 calibration points per subject, the fixation error can be reduced by 12.4% to just 1.11cm on smartphones and 1.59cm on tablets.

We note that this calibration phase is extremely simple and takes less than 1 minute for a 20 point calibration to complete. Since this is a one-time process, we see the overhead to be minimal. Figure 11 also implies that having a smaller number of calibration points (4) keeps the gaze estimation error low. Therefore, while our experiences suggest that 20 points meets a reasonable balance between calibration overhead and performance, this parameter can be adjusted with respect to the application goals.

5.7 Latency Improvements with Optical Flow

As discussed in Section 4.4, as a latency assist, we apply Lucas-Kanade optical flow tracking [39] to keep track of the eye landmarks (using 12 points) over consecutive frames. For each frame, if the tracked landmarks drift away from the reference landmarks in the previous frame by a certain distance (as measured by the Normalized Mean Error), we iterate through the entire gaze tracking pipeline, which includes the full face detection and facial landmark alignment operations (c.f., Fig. 1). Otherwise, *iMon* directly performs image enhancement and gaze estimation using the tracked landmarks on the input frames. Figure 12 shows the gaze tracking performance (frame error) for different frequencies of the optical flow detecting that the pre-processing can be skipped. For example, when 50% of frames bypass pre-processing, we see only a small error increase (to 1.51cm), and when processing one-third of incoming frames (i.e., 66% of frames skipped), the error is 1.55cm on mobile platforms. By skipping such frames we can omit the intermediate operations and minimize the processing latency. Overall, the plot suggests that optical flow can help significantly reduce the latency by avoiding unnecessary preprocessing with a small tracking error increase.

5.8 *iMon*'s Processing Latency on Mobile Platforms

We implemented *iMon* on iOS using the CoreML framework [27]. Except for the optical flow tracking component that uses OpenCV library APIs [5], all components are implemented as full-precision (float32) 'mlmodel' neural network models that run within the CoreML framework.

Table 6 shows the latency induced by each component of the *iMon* processing pipeline measured on an iPhone 12 Pro equipped with the Apple A14 chipset (4-core GPU and a 16-core Neural Engine specialized for neural network operations). Note that we run the eye-image enhancement model with a batch size of 2 for concurrently processing the left and right-eye images. The gaze estimation model is evaluated with single-sample inference. Additionally, we measured the latency of *iMon* when using the more latency-efficient MobileNetV2 as its core CNN module with optical flow enabled. When using the EfficientNetB3 model for gaze estimations,

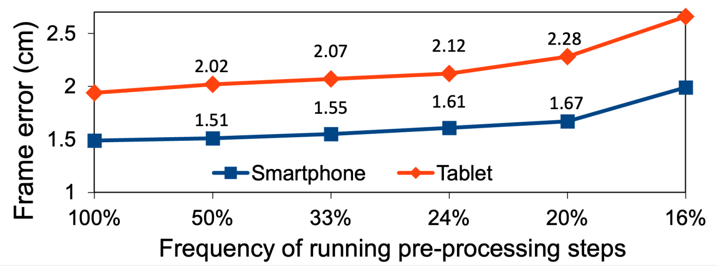


Fig. 12. Frame error for varying frequency of pre-processing operations when applying optical flow.

Table 6. Latency measurement of *iMon* and each of its components measured on an iPhone12 Pro.

Task	Latency
Face detection (S3fd)	9.36 ms
Facial landmark alignment (FAN)	9.07 ms
Optical flow tracking for eye landmarks	4.43 ms
Eye-image enhancement (UNet-ResNet18)	4.12 ms
Gaze estimation (EfficientNetB3)	56.63 ms
Gaze estimation (MobileNetV2)	2.14 ms
Full Tracking Pipeline	**Frame Rate**
iMon (EfficientNetB3) without optical flow	12.63 fps
iMon (EfficientNetB3) with optical flow	14.02 fps
iMon (MobileNetV2) without optical flow	40.50 fps
iMon (MobileNetV2) with optical flow	59.38 fps

we can notice from the table that its takes a (relatively) long latency to process each frame. Nevertheless, when optimizing for processing latency, by exploiting the MobileNetV2, *iMon* can perform real-time tracking at nearly 60fps with a person-independent per-frame tracking error of 1.63cm on smartphones and 2.34cm on tablets. This suggests that *iMon* is a mobile-suitable system that offers real-time and accurate gaze estimations.

5.9 Application-focused User Study

As the final part of our evaluation, we implemented a simple mobile application using *iMon* on the iPhone 12 Pro to perform a user study. The purpose of this study was to validate that *iMon* can be an effective tool for developing gaze-controlled applications by providing both accurate and timely gaze estimation results. Specifically, our IRB-approved user study first asks the participants to perform simple Kappa angle calibration and then to play a custom-designed ping-pong game in which the user controls the racket to hit a moving ball bouncing off the screen borders purely using eye movements as the control. I.e. *iMon* will track their eye movements in real-time and move the racket accordingly. Thus low latency and high accuracy are required for a good user experience.

Screenshots of these operations, both the calibration process and game are presented in Figure 13. The Kappa angle calibration requires the users to focus on the four corners and the center of the screen and visualize the real-time tracking results as a red circle (c.f., Fig 13 (a)). We use only five points for Kappa angle calibration given that our user study focused on only a single type of device orientation (i.e., vertical orientation). Once the calibration is done, the results will be used in the gaze estimation pipeline and our users will then be asked

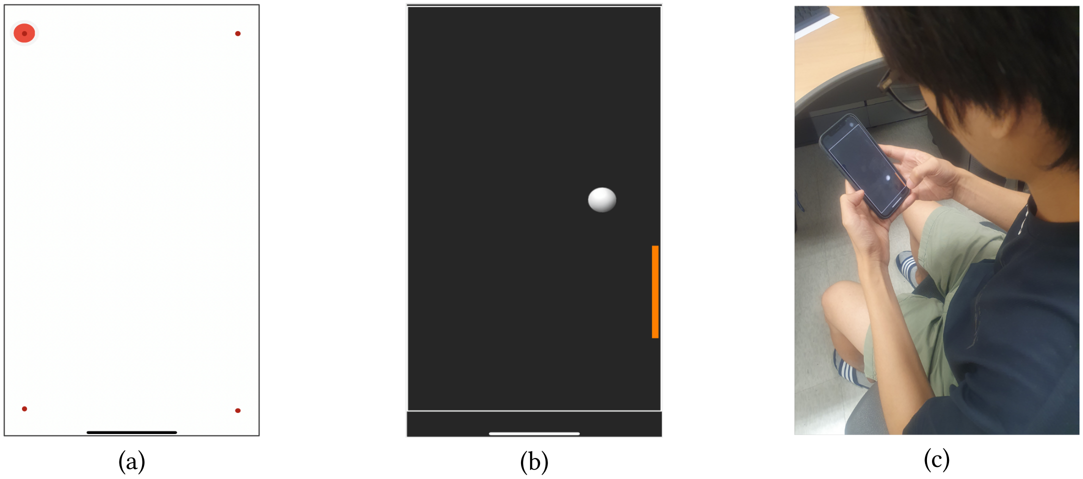


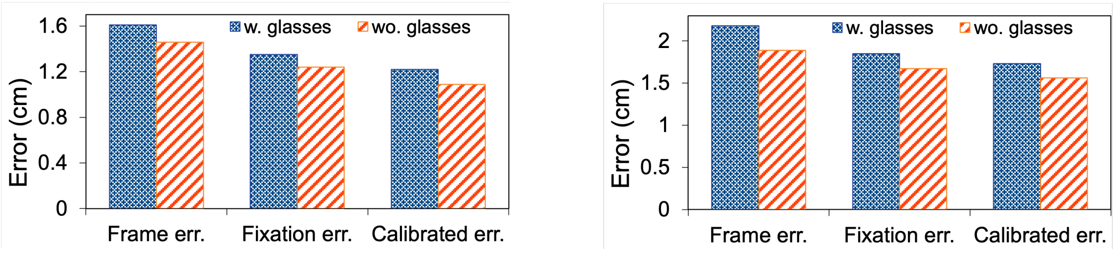
Fig. 13. Screenshots and picture of user study: (a) calibration phase screenshot: users are asked to perform a 5-point calibration (four corners and the center point of the screen) prior to starting the application; (b) ping-pong game app screenshot: the users control the orange bar (racket) with their gaze while the ball bounces off the screen edges; (c) picture of a participant playing the ping-pong game by controlling the racket using gaze movements.

Table 7. Survey question and results (on a 5-point Likert scale) from our user study. The frequency column shows the frequency of answer from the group providing fine-tuning data and the group of newly enrolled participants.

Survey question: *Please rate the quality of racket movements on a scale of 1-5*	Frequency with Fine-tuning	Frequency w/o Fine-tuning
(1) The racket moves randomly and/or does not seem to follow my gaze at all.	0 (0%)	0 (0%)
(2) The racket movement somewhat correlates with my gaze but the error is too high to play the game.	1 (14.2%)	0 (0%)
(3) I can move the racket most of the time using my gaze, but the error sometimes is still too high.	3 (42.9%)	2 (22.2%)
(4) The racket moves accordingly to my gaze with some tolerable error.	3 (42.9%)	7 (77.8%)
(5) The racket moves accordingly to my gaze perfectly.	0 (0%)	0 (0%)

to play 10 ping-pong game rounds. As Figure 13 (b) shows, users control the orange bar (racket) to bounce the moving white ball. For gameplay simplicity, we limit the racket to make only vertical movements. The racket size is 2.65 cm and the ball's radius is 0.34 cm. During gameplay, the ball bounces off the screen borders/walls in random directions and can land on any position on the 12.42 cm long right border of the screen, which is identical to the movable range of the racket. A video presenting how the user study is conducted is available at https://github.com/imonimwut/imon.

We enrolled 16 participants to the user study (age: 22.59 ± 3.31, 3 females, 8 wearing glasses) and randomly divided them into two groups – 7 participants in a group where model fine-tuning was performed and 9 participants in a group without fine-tuning. This allowed us to assess the benefits of fine-tuning *iMon* for each user. Each participant was asked to rate their gameplay experience, with respect to the racket movements controlled by the



(a) Smartphone

(b) Tablet

Fig. 14. Gaze tracking error evaluated on groups of participants wearing glasses compared to the group not wearing glasses in the GazeCapture test set.

gaze, through a single-question survey (with 5 options as listed in Table 7). The entire process took less than 10 mins to complete. The group that used model fine-tuning (comprising of 7 participants; 4 with glasses) provided 200 points of additional gaze tracking data to fine-tune the gaze estimation model. We note that all participants were temporarily asked to remove their masks during the study for effective facial landmark identification, and obeyed all COVID-19 related regulations.

Table 7 presents the results of the study for both groups of participants – those with and without model fine-tuning. As the results show, most participants indicated that the racket moved according to their gaze with tolerable error levels. Furthermore, we noticed that the difference in the gaze tracking errors of the two groups of participants was not significant. Overall, the user study suggests that *iMon* can be utilized as a general gaze tracking solution that can effectively handle real-time gaze control requests.

Quantitatively, the mean gaze tracking error across all the game sessions was 0.74 ± 0.43 cm – this was computed by averaging the distance between the racket and the ball center points at interaction events (when the ball hits the racket or the right wall/screen border). For the two groups, the computed error is 0.69 ± 0.39 cm and 0.78 ± 0.54 cm for the group that performed model fine-tuning and the group that did not, respectively. Furthermore, we quantify the performance of each participant group by counting the length of rallies in the last 7 rounds each user played in the ping-pong game (we removed the results from the first three game play rounds as the users needed to get used to playing the game). Specifically, the average rally length was 13.2 consecutive hits (max 38), and 9.7 (max 41) for the group with fine-tuning and the group without, respectively. This confirms that *iMon* can be an effective and easy-to-apply solution for supporting real-time gaze control mobile applications.

6 DISCUSSION & FUTURE WORK

• **Extending *iMon* to other mobile platforms:** We designed *iMon* to be a fast and accurate gaze tracking solution for mobile phones. As shown in Section 5, it outperforms prior work by up to 22% while still achieving real-time latencies (up to 60fps) on an iPhone 12 Pro. Since *iMon* uses deep learning models for many of its processing components, the implementation on latest Apple devices can leverage the Neural Engine designed to run machine learning operations efficiently. Therefore, the implementation of *iMon* on previous Apple device models (prior to iPhone 8) without Neural Engine would have significantly higher processing latency. For those devices, depending on the applications, one potential solution is to run *iMon* in a more latency-optimized configuration (using MobileNetV2 as backbone model and skip the image enhancement step). In the future, we also plan to extend *iMon* to Android by leveraging the Tensorflow Lite framework for fast on-device inference.

• **Improving the Performance of *iMon*:** There are a few orthogonal techniques that could be leveraged to improve the performance of *iMon*. First, Palmero et al. [47, 48] have demonstrated the potential of *integrating the temporal information* from sequences of eye images to improve the performance of appearance-based gaze tracking methods. However, spatio-temporal gaze estimation models usually leverage sequential components such as

Recurrent Neural Network or Long Short-Term Memory which could pose significant overheads. Therefore, special considerations/designs are required to make it practical for mobile platforms. Another important observation regarding the temporal aspect is that human eye movements can be categorized into a few types (e.g., saccades and pursuit movements) and each type has certain characteristics/constraints. For example, saccades movements have an involuntary fixed relationship between movement velocity, magnitude and duration. Those characteristics of eye movements could be helpful to design a more efficient tracking pipeline.

Next, Figure 11 shows that *personal calibration* can improve accuracy. However, a full calibration could take some time as the user has to focus on 20 random points. One possible solution to reduce this time is as users tend to look at the interacting point (e.g, text, button) before tapping, is to collect the calibration fixation data incrementally as users interact with their phone/applications – resulting in the tracking accuracy improving gradually as more opportunistic calibration data is collected.

Kraka et al [32] has demonstrated that *augmenting the data* by randomly shifting the eye regions within a certain range for each frame could improve the performance of gaze estimation models for both training and inference time phases. While data augmentation during inference time could be computationally costly and impractical on a mobile device, augmenting the training data could potentially improve the accuracy of *iMon* and also could be done offline on a separate server.

We also conducted an evaluation on the GazeCapture dataset to compare the performance of *iMon* between two groups of participants: with and without glasses. As Figure 14 shows, the tracking error of the group with glasses is consistently higher on both smartphones and tablets. On average, the tracking error of participants wearing glasses is 10% higher, even after calibration. Since each individual could have a different pair of glasses with certain shape, function and thickness, we think that this group represents a special case that requires additional data for gaze estimation model personalization.

Finally, we believe that fine-tuning *iMon* for each device model (e.g., specific models for an iPhone, or an Android device) would help to reduce the tracking error as each device has different characteristics (e.g., camera position, screen dimensions) that could directly affect the gaze estimation.

• **Generalizability:** Our user study in Section 5.9 was performed with a limited set of participants. While we could not actively recruit participants from a wide range of age groups due to restrictions from the COVID-19 pandemic, we believe that the participants that took part in the study (mostly in their 20's) represent a demographic that is most sensitive towards even small latency and accuracy issues that could occur when using gaze-based mobile control operations. Thus, we believe that our findings will hold even for a general population.

7 CONCLUSION

In this paper, we presented *iMon*, a highly accurate appearance-based gaze tracking system for mobile devices. *iMon* consists of a suite of three improvement techniques that significantly improve its accuracy compared to prior solutions. We first address the issue of low-fidelity ground truth gaze label data by proposing a 2D heatmap probabilistic representation for gaze labels. We then show how performing image enhancement to improve the visual quality and remove the motion blur effect of eye-region images can improve the overall accuracy. Finally, we apply a simple yet effective calibration scheme to mitigate the tracking error caused by the individual differences in Kappa angel. Overall, with all the improvements, *iMon*, compared to prior state-of-the-art results, improves the 2D gaze tracking accuracy by ∼22% when tested using the GazeCapture dataset and by ∼28% with the TabletGaze dataset. We also show through an IRB-approved user study that *iMon*'s gaze tracking performance can enable gaze-controlled applications with satisfying user acceptance levels. *iMon*'s source code, along with a video of it operating as part of the user study application is available at **https://github.com/imonimwut/imon**.

ACKNOWLEDGMENTS

This work was supported by the National Research Foundation of Korea (NRF) Grant funded by the Korean Government (MSIP) (No.2015R1A5A1037668, 2021R1A2C4002380).

REFERENCES

[1] Jason Antic, Jeremy Howard, and Uri Manor. 2019. Decrappification, DeOldification, and Super Resolution. https://www.fast.ai/2019/05/03/decrappify/.

[2] Saeed Anwar, Salman Khan, and Nick Barnes. 2019. A deep journey into super-resolution: A survey. *arXiv preprint arXiv:1904.07523* (2019).

[3] Tadas Baltrusaitis, Peter Robinson, and Louis-Philippe Morency. 2013. Constrained local neural fields for robust facial landmark detection in the wild. In *Proceedings of the IEEE international conference on computer vision workshops*. 354–361.

[4] Yiwei Bao, Yihua Cheng, Yunfei Liu, and Feng Lu. 2021. Adaptive feature fusion network for gaze tracking in mobile tablets. In *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 9936–9943.

[5] G. Bradski. 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools* (2000).

[6] Stephen L Brown and Miles Richardson. 2012. The effect of distressing imagery on attention to and persuasiveness of an antialcohol message: A gaze-tracking approach. *Health Education & Behavior* 39, 1 (2012), 8–17.

[7] Adrian Bulat and Georgios Tzimiropoulos. 2017. How far are we from solving the 2D & 3D Face Alignment problem? (and a dataset of 230,000 3D facial landmarks). In *International Conference on Computer Vision*.

[8] Alisa Burova, John Mäkelä, Jaakko Hakulinen, Tuuli Keskinen, Hanna Heinonen, Sanni Siltanen, and Markku Turunen. 2020. Utilizing VR and gaze tracking to develop AR solutions for industrial maintenance. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–13.

[9] Marcus Carter, Joshua Newn, Eduardo Velloso, and Frank Vetere. 2015. Remote gaze and gesture tracking on the microsoft kinect: Investigating the role of feedback. In *Proceedings of the Annual Meeting of the Australian Special Interest Group for Computer Human Interaction*. 167–176.

[10] Seung Ah Chung, Jaewon Choi, Seungchan Jeong, and Jeonggil Ko. 2021. Block-building performance test using a virtual reality head-mounted display in children with intermittent exotropia. *Eye (London, England)* 35, 6 (June 2021), 1758—1765. https://doi.org/10.1038/s41433-020-01160-y

[11] TensorFlow Core. [n.d.]. UpSampling2D. https://www.tensorflow.org/api_docs/python/tf/keras/layers/UpSampling2D.

[12] MS Corin, Teresita S Elizan, and Morris B Bender. 1972. Oculomotor function in patients with Parkinson's disease. *Journal of the neurological sciences* 15, 3 (1972), 251–265.

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 248–255.

[14] Mayada Elsabbagh, Evelyne Mercure, Kristelle Hudry, Susie Chandler, Greg Pasco, Tony Charman, Andrew Pickles, Simon Baron-Cohen, Patrick Bolton, Mark H Johnson, et al. 2012. Infant neural sensitivity to dynamic eye gaze is associated with later emerging autism. *Current biology* 22, 4 (2012), 338–342.

[15] SR Research EyeLink. [n.d.]. EyeLink Portable Duo. https://www.sr-research.com/eyelink-portable-duo/.

[16] Hamid Gharaee, Masoud Shafiee, Rafie Hoseini, Mojtaba Abrishami, Yalda Abrishami, and Mostafa Abrishami. 2015. Angle kappa measurements: normal values in healthy Iranian population obtained with the Orbscan II. *Iranian Red Crescent Medical Journal* 17, 1 (2015).

[17] Agostino Gibaldi, Mauricio Vanegas, Peter J Bex, and Guido Maiello. 2017. Evaluation of the Tobii EyeX Eye tracking controller and Matlab toolkit for research. *Behavior research methods* 49, 3 (2017), 923–946.

[18] Tianchu Guo, Yongchao Liu, Hui Zhang, Xiabing Liu, Youngjun Kwak, Byung In Yoo, Jae-Joon Han, and Changkyu Choi. 2019. A Generalized and Robust Method Towards Practical Gaze Estimation on Smart Phone. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 0–0.

[19] Dan Witzner Hansen and Qiang Ji. 2009. In the eye of the beholder: A survey of models for eyes and gaze. *IEEE transactions on pattern analysis and machine intelligence* 32, 3 (2009), 478–500.

[20] D. W. Hansen and Q. Ji. 2010. In the Eye of the Beholder: A Survey of Models for Eyes and Gaze. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32, 3 (2010), 478–500. https://doi.org/10.1109/TPAMI.2009.30

[21] Corinna Haupt, Andrea B Huber, et al. 2008. How axons see their way–axonal guidance in the visual system. *Front Biosci* 13 (2008), 3136–3149.

[22] Junfeng He, Khoi Pham, Nachiappan Valliappan, Pingmei Xu, Chase Roberts, Dmitry Lagun, and Vidhya Navalpakkam. 2019. On-device few-shot personalization for real-time gaze estimation. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 0–0.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[24] Qiong Huang, Ashok Veeraraghavan, and Ashutosh Sabharwal. 2017. TabletGaze: dataset and analysis for unconstrained appearance-based gaze estimation in mobile tablets. *Machine Vision and Applications* 28, 5-6 (2017), 445–461.

[25] Sinh Huynh, Rajesh Krishna Balan, JeongGil Ko, and Youngki Lee. 2019. VitaMon: Measuring Heart Rate Variability Using Smartphone Front Camera. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems* (New York, New York) *(SenSys '19)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3356250.3360036

[26] iMotions. [n.d.]. Tobii Pro X2-60. https://imotions.com/hardware/tobii-x2-60/.

[27] Apple Inc. 2021. Core ML - Apple Developer Documentation. https://developer.apple.com/documentation/coreml.

[28] Vidit Jain and Erik Learned-Miller. 2010. *Fddb: A benchmark for face detection in unconstrained settings.* Technical Report. UMass Amherst technical report.

[29] Harini D Kannan. 2017. *Eye tracking for the iPhone using deep learning.* Ph.D. Dissertation. Massachusetts Institute of Technology.

[30] Tero Karras, Samuli Laine, and Timo Aila. 2019. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4401–4410.

[31] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[32] Kyle Krafka, Aditya Khosla, Petr Kellnhofer, Harini Kannan, Suchendra Bhandarkar, Wojciech Matusik, and Antonio Torralba. 2016. Eye tracking for everyone. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2176–2184.

[33] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 1097–1105. http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[34] Amit Laddi and Neelam Rup Prakash. 2019. Eye gaze tracking based directional control interface for interactive applications. *Multimedia Tools and Applications* 78, 22 (2019), 31215–31230.

[35] Hsin-Yu Lai, Gladynel Saavedra-Pena, Charles G Sodini, Vivienne Sze, and Thomas Heldt. 2019. Measuring Saccade Latency using Smartphone Cameras. *IEEE Journal of Biomedical and Health Informatics* 24, 3 (2019), 885–897.

[36] Michael Lankes and Andreas Haslinger. 2019. Lost & found: Gaze-based player guidance feedback in exploration games. In *Extended Abstracts of the Annual Symposium on Computer-Human Interaction in Play Companion Extended Abstracts*. 483–492.

[37] Joseph Lemley, Anuradha Kar, Alexandru Drimbarean, and Peter Corcoran. 2019. Convolutional neural network implementation for eye-gaze estimation on low-quality consumer imaging systems. *IEEE Transactions on Consumer Electronics* 65, 2 (2019), 179–187.

[38] Feng Lu, Yusuke Sugano, Takahiro Okabe, and Yoichi Sato. 2014. Adaptive linear regression for appearance-based gaze estimation. *IEEE transactions on pattern analysis and machine intelligence* 36, 10 (2014), 2033–2046.

[39] Bruce D Lucas, Takeo Kanade, et al. 1981. An iterative image registration technique with an application to stereo vision. (1981).

[40] Susana Martinez-Conde, Stephen L Macknik, and David H Hubel. 2004. The role of fixational eye movements in visual perception. *Nature reviews neuroscience* 5, 3 (2004), 229–240.

[41] Carlos Hitoshi Morimoto, Dave Koons, Arnon Amir, and Myron Flickner. 2000. Pupil detection and tracking using multiple light sources. *Image and vision computing* 18, 4 (2000), 331–335.

[42] Carlos H Morimoto and Marcio RM Mimica. 2005. Eye gaze tracking techniques for interactive applications. *Computer vision and image understanding* 98, 1 (2005), 4–24.

[43] Samuel Arba Mosquera, Shwetabh Verma, and Colm McAlinden. 2015. Centration axis in refractive surgery. *Eye and Vision* 2, 1 (2015), 4.

[44] Takashi Nagamatsu, Yukina Iwamoto, Junzo Kamahara, Naoki Tanaka, and Michiya Yamamoto. 2010. Gaze estimation method based on an aspherical model of the cornea: surface of revolution about the optical axis of the eye. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*. ACM, 255–258.

[45] Kristin R Newman and Christopher R Sears. 2015. Eye gaze tracking reveals different effects of a sad mood induction on the attention of previously depressed and never depressed women. *Cognitive Therapy and Research* 39, 3 (2015), 292–306.

[46] Jorge Otero-Millan, Xoana G Troncoso, Stephen L Macknik, Ignacio Serrano-Pedraza, and Susana Martinez-Conde. 2008. Saccades and microsaccades during visual fixation, exploration, and search: foundations for a common saccadic generator. *Journal of vision* 8, 14 (2008), 21–21.

[47] Cristina Palmero, Javier Selva, Mohammad Ali Bagheri, and Sergio Escalera. 2018. Recurrent cnn for 3d gaze estimation using appearance and shape cues. *arXiv preprint arXiv:1805.03064* (2018).

[48] Cristina Palmero Cantarino, Oleg V Komogortsev, and Sachin S Talathi. 2020. Benefits of temporal information for appearance-based gaze estimation. In *ACM Symposium on Eye Tracking Research and Applications*. 1–5.

[49] HyeonJung Park, Youngki Lee, and JeongGil Ko. 2021. Enabling Real-Time Sign Language Translation on Mobile Platforms with On-Board Depth Cameras. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 5, 2, Article 77 (June 2021), 30 pages. https://doi.org/10.1145/3463498

[50] Jim R Parker and AQ Duong. 2009. Gaze tracking: A sclera recognition approach. In *2009 IEEE International Conference on Systems, Man and Cybernetics*. IEEE, 3836–3841.

[51] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 234–241.

[52] Michele Rucci, Ramon Iovin, Martina Poletti, and Fabrizio Santini. 2007. Miniature eye movements enhance fine spatial detail. *Nature* 447, 7146 (2007), 852–855.

[53] Christos Sagonas, Georgios Tzimiropoulos, Stefanos Zafeiriou, and Maja Pantic. 2013. 300 faces in-the-wild challenge: The first facial landmark localization challenge. In *2013 IEEE International Conference on Computer Vision Workshops*. IEEE, 397–403.

[54] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4510–4520.

[55] Akanksha Saran, Srinjoy Majumdar, Elaine Schaertl Short, Andrea Thomaz, and Scott Niekum. 2018. Human gaze following for human-robot interaction. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 8615–8621.

[56] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[57] Junbong Song, Sungmin Cho, Seung-Yeob Baek, Kunwoo Lee, and Hyunwoo Bang. 2014. GaFinC: Gaze and Finger Control interface for 3D model manipulation in CAD application. *Computer-Aided Design* 46 (2014), 239–245.

[58] Kar-Han Tan, David J Kriegman, and Narendra Ahuja. 2002. Appearance-based eye gaze estimation. In *Sixth IEEE Workshop on Applications of Computer Vision, 2002.(WACV 2002). Proceedings*. IEEE, 191–195.

[59] Mingxing Tan and Quoc V Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946* (2019).

[60] Ying-li Tian, Takeo Kanade, and Jeffrey F Cohn. 2000. Dual-state parametric eye tracking. In *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)*. IEEE, 110–115.

[61] Haiyuan Wu, Yosuke Kitagawa, Toshikazu Wada, Takekazu Kato, and Qian Chen. 2007. Tracking iris contour with a 3D eye-model for gaze estimation. In *Asian Conference on Computer Vision*. Springer, 688–697.

[62] Shuo Yang, Ping Luo, Chen-Change Loy, and Xiaoou Tang. 2016. Wider face: A face detection benchmark. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 5525–5533.

[63] Shifeng Zhang, Xiangyu Zhu, Zhen Lei, Hailin Shi, Xiaobo Wang, and Stan Z Li. 2017. S3fd: Single shot scale-invariant face detector. In *Proceedings of the IEEE International Conference on Computer Vision*. 192–201.

[64] Xiaoyi Zhang, Harish Kulkarni, and Meredith Ringel Morris. 2017. Smartphone-based gaze gesture communication for people with motor disabilities. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 2878–2889.

[65] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. 2015. Appearance-based gaze estimation in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4511–4520.

[66] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. 2017. It's written all over your face: Full-face appearance-based gaze estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 51–60.

[67] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. 2017. Mpiigaze: Real-world dataset and deep appearance-based gaze estimation. *IEEE transactions on pattern analysis and machine intelligence* 41, 1 (2017), 162–175.