An End-to-End Review of Gaze Estimation and its Interactive Applications on Handheld Mobile Devices

YAXIONG LEI, University of St Andrews, UK
SHIJING HE, King’s College London, UK
MOHAMED KHAMIS, University of Glasgow, UK
JUAN YE, University of St Andrews, UK

In recent years we have witnessed an increasing number of interactive systems on handheld mobile devices which utilise gaze as a single or complementary interaction modality. This trend is driven by the enhanced computational power of these devices, higher resolution and capacity of their cameras, and improved gaze estimation accuracy obtained from advanced machine learning techniques, especially in deep learning. As the literature is fast progressing, there is a pressing need to review the state of the art, delineate the boundary, and identify the key research challenges and opportunities in gaze estimation and interaction. This paper aims to serve this purpose by presenting an end-to-end holistic view in this area, from gaze capturing sensors, to gaze estimation workflows, to deep learning techniques, and to gaze interactive applications.

CCS Concepts: · General and reference → Surveys and overviews; · Computing methodologies → Computer vision tasks; · Human-centered computing → Ubiquitous and mobile computing systems and tools; Interactive systems and tools; Interaction techniques.

Additional Key Words and Phrases: Gaze Estimation, Domain Adaptation, Datasets, Eye Tracking, Eye Movement, Gaze Data Process, Gaze Interaction Methods, Gaze-based Interaction, Smartphones, Handheld Mobile Devices

1 INTRODUCTION

Gaze interaction is to make use of gaze to facilitate interactions with computing devices, including virtual reality (VR) headsets, mobile or wearable devices, desktops, and robots. Gaze refers to a point on a screen or a direction in space and can be inferred from pupil positions, facial structure, and head movements. Through gaze, a system can sense users’ attention [233] and enable touch-free interaction, which has driven a wide range of applications.

In a simulated surgical training task, gaze has been used to aid the identification of a target on a subject’s laparoscopic screen in order to reduce mistakes and overcome language barriers [32]. In a driving situation, real-time gaze coding allows to track a driver’s attention and detecting whether they are distracted or fatigue [106, 255], and support automated driving [170]. Gaze can also help diagnose mental health or autism by analysing the scan path [236]. Moving beyond, the gaze is also presented as a human-computer interface, facilitating controlling Internet of Things (IoT) devices in a smart home system [127], and empowering people with a physical impairment to interact with applications such as creative art [49, 140, 263].

The advanced research in gaze interaction has inspired an increasing number of industrial applications, including optimising the system’s scheduling of processing resources [166], facilitating the presentation of
content [59, 126], marketing, and accessibility for people with disabilities [34, 58, 88, 267]. Smart Eye brings an AI-integrated eye-tracking technology to detect whether a driver is distracted [221], and Eyetech Digital Systems launched EyeOn to enable users to type and communicate using eye movement [55]. There are growing market players, including Tobii AB, LC Technologies, Eyetech Digital Systems, Ergoneers GmbH, Smart Eye AB., Mirametrix Inc., Pupil Labs GmbH, Seeing Machines, SR Research Ltd., and Gazepoint. The worldwide market for eye tracking technology is valued at USD 638.8 million and is anticipated to grow at a compound annual growth rate (CAGR) of 33.4% between 2022 and 2030 [76].

In recent years, mobile devices have advanced with significantly improved camera quality and computational power. These capabilities make it possible to explore gaze interaction on mobile devices. The advantage over traditional eye tracking is that gaze on mobile devices is derived from images or videos on cameras, which does not require extra devices. Industries are starting to explore novel gaze interactions on mobile devices. For example, Apple has paired facial recognition and eye movements to enhance the unlocking experience of FaceID and the Huawei Mate 40 Pro keeps the screen on when being gazed at.

What currently needs to be added to the literature is a comprehensive overview of gaze estimation to interaction on handheld mobile devices and a roadmap for advancing from low-level gaze points and directions to high-level gaze patterns, which play a significant role in many gaze applications. To close the gap, we describe the current technological advances in capturing gaze and a workflow along with deep learning algorithms to estimate gaze. We identify the challenges in enhancing the diversity of data collection and robustness of gaze estimation, especially reflecting the complexities of dynamic environments such as partial faces, varying lighting conditions, and constant changes of holding postures. These challenges come from the inherent characteristics of mobile devices: their size and mobility, which fundamentally differ from eye tracking on other platforms such as VR, desktops, and large displays. We also broadly review gaze-based applications, introduce eye physiology, and point to new types of gaze for future interaction design. Different from the existing surveys on gaze estimation, this paper makes the following key contributions.

- We present an end-to-end holistic review from sensors, to algorithms, and to application.
- We focus on appearance-based gaze estimation on handheld devices, and identify the unique challenges different from other platforms.
- We bridge the gap between gaze estimation, gaze data processes, and gaze interaction and present a pipeline for processing, analysing, and deriving high-level gaze events from raw gaze data.

2 METHODOLOGY

Our paper takes recent reviews and survey studies as an initial research methodology [31, 91, 111, 116, 118], with adaptation to suit our research aims; that is, understanding the workflow of gaze interaction from camera input to applications on handheld mobile devices. Our research methodology comprises four stages: defining keywords, paper collection, paper categorisation and collation, and paper analysis. We extract keywords based on the content in the papers mentioned in the related work, and eventually identify (“Eye” or “Gaze” or “Eye Tracking”) and (“Gaze Estimation” or “Eye Tracking Technique” or “Eye Tracking algorithm” or “Algorithm” or “Dataset”) for Section 4 and 5, (“Eye” or “Gaze” or “Eye Tracking” or “Eye Movement”) and (“Data Analysis” or “Data Process” or “Application” or “Interaction”) for Section 6 and 7. We select the conferences and journals in the areas of HCI, UbiComp, computer vision, and eye-tracking as our target sources, including ETRA, CHI, IJHCS, HCI, UbiComp/IMWUT, MobileHCI, IJHCI, CVPR, EMR, BRM, etc. We also search terms on Google Scholar to catch more publications.

We categorise papers based on research topics: gaze estimation algorithms, gaze data analysis, gaze interaction and gaze applications. We then sub-categorise them according to the platforms in which these items are applied, e.g., handheld mobile devices (mobile phones or tablets), desktop devices (desktop computers or laptops), and
head-mounted devices (VR or glasses). Algorithms, interactions and applications that are currently used in various
devices will be discussed in terms of their applicability to handheld mobile devices. Our discussion focuses on
the characteristics of handheld mobile devices, i.e. the mobility of the device, the screen size, the application
characteristics, and the usage scenario.

Based on the classification and year, we summarise the content of the collected papers, extract their motivation,
methodology, subsequent research directions and their contribution to gaze estimation algorithms and applications
for handheld mobile devices.

3 RELATED WORK

In recent years, more and more research projects on gaze are emerging, addressing the quest for performance
and stability in gaze estimation, gaze interaction and gaze applications using the capabilities of commercial
eye trackers or gaze estimation algorithms. The existing surveys and review studies have different perspectives
and areas: algorithms and applications of gaze estimation. Kar and Corcoran [111] survey nearly 20 years
of gaze estimation research on a variety of devices, including desktop, television, head-mounted, automotive
and handheld, and also focus on methods for evaluating the performance of gaze tracking systems. Cheng et
al. [31] provide a detailed review of appearance-based gaze estimation approaches. They build a pipeline of gaze
estimation of deep learning. In terms of applications, Khamis et al. [118] summarise eye-tracking related studies
on handheld mobile devices from 2002 to 2018. They list three main topics of gaze applications: gaze behaviour
analysis, implicit gaze interaction and explicit gaze interaction. Katsini et al. [116] focus on the application of gaze
in security and privacy, which include authentication, privacy protection and gaze monitoring during security
critical tasks. They summarise the usage scenarios, devices and evaluation methods for these tasks. Hirzle et
al. [91] conducted a study on existing gaze interaction projects around the Digital Eye Strain (DES), which has
surveyed over 400 papers published in the last 46 years. Ghosh et al. [74] focus on the algorithms in AR/VR
devices and the applications in healthcare and driver engagement. Nishan et al. [77] provides a detailed analysis
of eye-tracking technology on mobile devices and present an edge computing solution for real-time eye-tracking
experience.

Different from the above reviews, our paper aims to present an end-to-end overview of gaze estimation, data
analysis and interaction, illustrating its workflow, mainstream deep learning techniques for estimating gaze, and
a broad range of gaze interaction applications on handheld mobile devices.

4 GAZE ESTIMATION ON HANDHELD MOBILE DEVICES

This section presents an overview of gaze estimation process, including the task of gaze estimation (in Section 4.1),
sensors common in handheld mobile devices for gaze estimation (in Section 4.2), and the workflow (in Section 4.3)
including pre-processing, learning, post-processing, and calibration.

4.1 Task and Evaluation of Gaze Estimation

Gaze estimation refers to predicting a point of gaze (PoG) [31] from images or videos captured on the front
camera of handheld mobile devices, as presented in Figure 1; i.e., a coordinate point \( p = (u, v) \) on a screen; or
gaze direction; i.e., a vector \( g = (g_x, g_y, g_z) \) in a 3D coordinate system [171]. The former is often referred to as 2D
gaze while the latter as 3D gaze. In both cases, it is treated as a regression task, while it can also be treated as a
classification problem [240]; that is, predicting which grid or area the user is looking at.

The accuracy of gaze estimation is defined as the average difference between the estimated gaze location and
the location of the fixation target. In 2D, the difference is measured as the Euclidean distance between the true
point \( p \) and estimated point \( \hat{p} \), where a point is commonly represented as a 2D coordinate in pixels with units of

ACM Comput. Surv.
The gaze estimation task involves predicting either a) a point of gaze (PoG) as a coordinate point on a screen (called 2D gaze) or b) a gaze direction in a 3D space (called 3D gaze) [31].

The true points can be obtained via an extra high-resolution eye-tracking device or self-reported by participants; that is, indicate their gaze points using a cursor or touch [98, 108, 114, 156, 246]. For example, GazeCapture [138] designs an application to display a red dot on a screen of a mobile phone and requires participants to gaze at the dot and follow its movement. Before the dot moves, a small letter L or R is displayed for 0.05 seconds, which requires a participant to tap either the left (L) or right (R) side of the screen. This serves as a way to monitor participants’ attention and thus validate the data collection. The ground truth points are the locations of the red dot.

In 3D, the difference in gaze direction refers to an angular distance between the true direction vector $\mathbf{g}$ and the estimated direction $\hat{\mathbf{g}}$, which is often represented in degrees.

$$\text{acc}_{3D} = \alpha^\circ = \frac{\mathbf{g} \cdot \hat{\mathbf{g}}}{\|\mathbf{g}\|\|\hat{\mathbf{g}}\|}$$

The collection of gaze direction is more complex, as the ground truth is a vector in a 3D space. Gaze360 [117] uses a 360° panoramic camera placed on a tripod in the centre of the scene, and a large moving rigid target board marked with an AprilTag and a cross on which participants are instructed to continuously fixate. The true gaze direction is derived from the camera coordinator system based on the distance between the eye, the camera, and the target board. Differently, ETH-XGaze [268] utilises 18 single-lens reflex cameras in a 3D space to gather ground truth while stimuli are displayed on a large screen in front of participants.

Commercial eye-tracking technologies like Tobii [229] also provide the other two measures: precision and data loss. Precision refers to the system’s ability to produce the same gaze point or direction measurement reliably. It is calculated as the root mean square of accuracy on a sequence of consecutive data pairs of true and estimated gazes.

$$\text{precision} = \sqrt{\frac{\sum_{i=1}^{n} \text{acc}_i^2}{n}},$$

where $\text{acc}_i$ is the accuracy (2D or 3D) on the $i$th pair between the target location and the gaze (point or direction), referring to the above Equation (1) and (2).
Data loss is defined as the ratio of gaze samples captured by the eye tracker during the fixation on a target. It is calculated by excluding invalid samples; that is, no gaze is detected.

\[
data_{loss} = \frac{\text{No. of invalid samples}}{\text{No. of total samples}}
\]  

(4)

Current gaze estimation algorithms typically focus on providing accuracy metrics and do not include precision and data loss measurements. However, for practical applications, it is important to consider all the above three metrics and provide them in their test report to ensure reliable and stable performance in real-world applications.

4.2 Sensors

Eye-tracking technologies have advanced significantly over the past 60 years [215], from electro-oculography (EOG) signals that are based on muscle action signals to visual signals from cameras. The recent examples include head-mounted devices [18] like Oculus VR or HoloLens 2, wearable eye trackers like Tobii Pro Glasses 3, and various types of screen-based eye trackers [185]. These devices come in a variety of shapes but generally tightly integrate the camera above or below a screen; for example, a camera at the top of a mobile phone. They have different eye-to-screen distances; for example, head-mounted eye trackers are close to the eye (e.g., within the distance of 0.5-3 cm), while in the screen-based eye trackers (aka remote eye trackers), the distance is usually 50-80 cm for desktop settings [195], and 15-50 cm for handheld devices [100, 232]. The distance and the screen size of handheld devices may constrain what types of gaze interaction applications are more acceptable to users. In this survey, we focus on sensors on handheld mobile devices, which are RGB, RGB-D (depth), and Infrared (IR) cameras (see Figure 2).

4.2.1 RGB Camera. Currently, most mobile device-based gaze estimation methods are using RGB Cameras, which are widely supported. Over years, the quality of the cameras has improved, in terms of aperture, resolution, and frame rate.

4.2.2 RGB-D Camera. RGB-D camera adds depth information to RGB images; that is, each pixel relates to a distance between the object in the image and the image plane. RGB-D cameras are starting to be supported in modern smartphones, especially high-end phones. Depth information can assist in constructing a 3D model of head pose and provide information on the eye position, further improving gaze estimation [171, 253].

4.2.3 IR Camera. IR camera refers to a Near Infrared (NIR) camera [196] that uses an artificial IR light source aimed on- and off-axis at the eye, which introduces glint, called corneal reflection [161]. The glint acts as a reference point and the gaze direction is calculated by measuring the changing relationship between the glint...
and the moving pupil centre of the eye. IR camera is sensitive to wavelengths from 700 nanometers (nm) to 1,400 nm, and this band does not affect human vision beyond the range of visible spectrum [225], so it is often used as near-eye cameras; e.g., head-mounted augmented and virtual reality (AR/VR) devices [252]. Recently, there are more and more mobile phones that are configured with NIR cameras, including iPhone, Huawei, Samsung, Xiaomi and OPPO.

4.3 Workflow

Fig. 3. Workflow of gaze estimation based on camera

As presented in Figure 3, the workflow of gaze estimation on handheld mobile devices consists of four main stages: (1) pre-processing – rectifying and processing images or videos acquired from a camera to identify a face and extract features; (2) learning – applying a machine learning technique to estimate a gaze point or vector; (3) post-processing – converting the gaze output to suit the requirements of applications; and (4) calibration – calibrating the gaze estimation model to cater for the characteristics and context of a new environment, device, and user. In the following, we will illustrate each of the above stages.

4.3.1 Pre-processing and Feature Extraction. At the pre-processing stage, images are collected from a camera, from which a face can be identified and facial landmarks are extracted. Facial landmarks, referred to as a set of coordinates on an image, are used to locate and represent important regions of the face, including the chin, mouth, nose, eyes, and eyebrows. OpenFace [12] and Dlib [128] are the most commonly used libraries for face
identification and landmark extraction [154, 271, 272]. Recently, researchers are starting to design customised networks for more accurate and richer facial feature extraction, including Multitask Cascaded Convolutional Network (MTCNN) [264], Deep Alignment Network (DAN) [136], Position Map Regression Network (PRNet) [60], and 3D Dense Face Alignment (3DDFA) [279]. Significant effort has also been dedicated to pupil segmentation and detection, including PupilNet [67], Circular Binary Features (CBF) [64] and Boosted-Oriented Edge optimisation (BORE) [63] for real-time pupil detection.

Once facial landmarks are extracted, face and/or eye images can be cropped. A common way to do so is to crop a square region with the centre of the face and a width. The centre of the face is the averaged coordinate positions of all the facial landmarks and the width can be set as a ratio to the maximum distance between the landmarks; for example, Zhang et al. [272] set the ratio as 1.5. As head pose has a significant impact on gaze estimation, often the images need to be rectified, including rotating and shifting to align with a reference point [31]. With cropped face and eye images, various features can be extracted, including eye, face, face and eye, and temporal.

**Eyes.** The eye is intrinsically connected to gaze estimation, as any change in the gaze direction leads to a corresponding alteration in the eye’s appearance. For instance, eye rotation affects the iris’s position and the eyelid’s shape, resulting in a shift in the gaze direction. This connection enables gaze estimation based on the eye’s appearance. Many methods leverage both eyes as cascading features for gaze estimates, with examples including, Minst [271], SAGE [82], GoogleGaze [232], EyeNet [185], and EVE-SCPT [13]. However, for gaze estimation from handheld mobile devices, the visibility of both eyes only has 68.18% [119], so monocular features remain valuable and are used by DPGE [187], Deng et al. [278] and OneEye-Net [5].

**Face.** Facial images provide valuable information on the head pose, which is beneficial for gaze estimation. Numerous methods utilise facial features as input for their models, such as FullFace [272], GazeTR [29], L2CS-Net [1], GazeCLR [109], SE-Gaze [178], FreeGaze [44], GazeNAS [174], and Gaze360 [117]. However, facial images can also contain redundant information. Researchers, as seen in PureGaze [28] and other models [179, 270, 272], have attempted to filter out irrelevant features in facial images, enabling the model to focus on the core facial features that are common to all. This optimisation mechanism implicitly eliminates gaze-irrelevant features, thus enhancing the gaze estimation network’s robustness. Additionally, facial landmarks are employed as supplementary features to model head pose and eye position.

**Face and Eyes.** In order to have robust features, some studies have employed both eye and face features as input to have robust features, with these networks typically being multi-streamed to obtain information from the face and eyes simultaneously. Facial landmarks and face grid are also utilised as additional features to model for head pose, eye position and spatial information, often used in conjunction with eye and face features. Examples of such methods include iTTracker [138], GazeAttentionNet [96], Dilated-Net [26], TAT [78], AFF-Net [14], iMon [102], RecurrentGaze, [181], GAZEL [184], RT-Gene [62], and HAZE-Net [258].

**Static and Temporal Features.** Features obtained from images are static; however, the information captured by a camera is dynamic, with a correlation between frames. To improve the robustness on prediction, some studies [102, 117, 181, 185] investigate the temporal information by extracting features from video and learning their temporal correlations between frames. Temporal features, such as optical flow [102] and eye movement dynamics [241], have been used to improve gaze estimation accuracy. Optical flow provides information about motion between frames. Gaze360 [117], RecurrentGaze [181], EyeNet [185] and EVE-SCPT [13] directly apply models such as LSTM to obtain correlation information between video frames.

**4.3.2 Learning.** Table 1 and Figure 4 present the mainstream deep learning models applied to gaze estimation, which are built on Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and more recently Transformers.
Convolutional Neural Networks. CNN is one of the most commonly used techniques for extracting facial and eye features. In gaze estimation, VGG [62, 218, 271, 273], ResNet [83, 117, 268], LeNet [146], AlexNet [141], MobileNet-V2 [25] and DenseNet [94] are the mostly adopted state-of-the-art architectures. Researchers also design novel architectures for face and eye feature extraction. Zhang et al. [272] have proposed a customised CNN with spatial weights for full-face 2D and 3D gaze estimation. The first few layers of the CNN extract image features, and then the last few layers are dedicated to learning spatial weights for the activation of the last convolutional layer. The purpose of the spatial weights is to indicate the importance of different regions of the face for gaze estimation.
Features from eye images can be combined with the head pose, and the eye features can be explored independently (with each of the two eyes) or jointly (with two eyes together). Fischer et al. [62] have applied two VGG-16 networks [218] to learn individual features from two eye images and then concatenate these features to predict the yaw and pitch gaze angles. Cheng et al. [30] have proposed an asymmetric regression evaluation network where a four-stream CNN is applied to extract and combine features on individual eye images, which are used in linear regression to estimate the 3D gaze direction.

Recurrent Neural Networks. RNN [208] and Long short-term memory (LSTM) [92] have been employed to explore and leverage the temporal relationships between consecutive frames to improve the estimation accuracy. Palmero et al. [181] have proposed a Recurrent CNN Regression Network for 3D gaze estimation. The network is composed of 3 modules: individual, fusion, and temporal. The individual module learns features from each appearance cue independently and consists of a two-stream CNN: one for processing face images and the other for jointly learning eye images. The fusion module combines the extracted features of each appearance stream in a single vector along with the normalised landmark coordinates. Then, it learns a joint representation between modalities in a late-fusion fashion. Both individual and fusion modules are applied to each frame, and the resulting feature vectors of each frame are input to the temporal module based on a many-to-one recurrent network. This module leverages sequential information to predict the normalised 3D gaze angles of the last frame of the sequence using a linear regression layer.

Bidirectional LSTMs are another popular choice for modelling temporal relationships between successive frames [117, 277]. Kellnhofer et al. [117] have utilised 7 consecutive sequence frames to obtain continuous head and eye movements in the video stream to improve the accuracy of predicting the gaze of the central frame in the input sequence. Specifically, a head crop from each frame is individually processed by ResNet-18, and the extracted features are fed to a bidirectional LSTM with two layers that take the sequence of forward and backward vectors. Finally, these vectors are concatenated and passed through a fully connected layer to produce two outputs: gaze prediction and error quantile estimation.

Combining the gaze direction with the optical low in the eye region is another way of exploiting temporal information in motion. Bace et al. [8] have combined gaze direction with the optical low in the eye region to identify the target that a user is following. The system consists of two components: one for taking a facial image as input and predicting 2D gaze points, and the other for estimating the motion pattern between consecutive frames. Aggregating the outputs from these two components can improve the robustness of estimation.

Transformers. Transformer, originating from natural language processing [234], contains self-attention layers, layer normalisation and multi-layer perception layers. Compared to RNN, self-attention layers have global computations and better memory to process long-sequence tasks. Transformers also perform well in computer vision tasks by replacing words in NLP tasks with non-overlapping image patches, called vision transformer (ViT) [39]. GazeTR [29] is an early ViT adopter in the field of gaze estimation. They apply the original ViT architecture to gaze estimation and also propose a hybrid ViT architecture combined with a CNN feature extractor. The hybrid ViT architecture achieved better results than ViT alone on various datasets. GazeNAS [174] is another early adaptor of ViT, where they propose a light-weighted ViT that only has 1 million parameters and uses 0.28 GFLOPs. It involves neural architecture search (NAS) from reinforcement learning as a multi-resolution feature extractor. GazeNAS achieves state-of-the-art results on various benchmark datasets.

Semi-supervised and Unsupervised Learning. Most deep learning algorithms require a large number of training data and collecting high-quality gaze datasets is a time- and effort-consuming task. Therefore, in recent years, we have witnessed more and more semi-supervised and unsupervised learning techniques being developed to reduce the reliance on labelled data. Kothari et al. [134] have proposed to curate videos from the Web where people are “looking at each other” (LAEO), and annotate each frame with whether LAEO is present. They design
Table 1. A list of representative appearance-based gaze estimation models

<table>
<thead>
<tr>
<th>Methods</th>
<th>Attributes</th>
<th>Year</th>
<th>Feature</th>
<th>Model</th>
<th>Dataset</th>
<th>PoG(cm)</th>
<th>Direction(°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minst [271]</td>
<td>Eyes, Lmks</td>
<td>2015</td>
<td>CNN</td>
<td>MPIIGaze</td>
<td>-</td>
<td>-</td>
<td>6.27°</td>
</tr>
<tr>
<td>iTracker [138]</td>
<td>Face, Eyes, Grid</td>
<td>2016</td>
<td>CNN</td>
<td>GazeCapture</td>
<td>2.81 1.86</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>FullFace [272]</td>
<td>Face</td>
<td>2017</td>
<td>CNN</td>
<td>MPIIFaceGaze</td>
<td>-</td>
<td>-</td>
<td>4.80°</td>
</tr>
<tr>
<td>Dilated-Net [26]</td>
<td>Face, Eyes</td>
<td>2018</td>
<td>CNN</td>
<td>MPIIGaze</td>
<td>-</td>
<td>-</td>
<td>5.12°</td>
</tr>
<tr>
<td>RT-Gene [62]</td>
<td>Face, Eyes</td>
<td>2018</td>
<td>CNN</td>
<td>MPIIGaze</td>
<td>-</td>
<td>-</td>
<td>4.66°</td>
</tr>
<tr>
<td>DPGE [187]</td>
<td>Eye</td>
<td>2018</td>
<td>CNN</td>
<td>MPIIGaze</td>
<td>-</td>
<td>-</td>
<td>4.50°</td>
</tr>
<tr>
<td>RecurrentGaze [181]</td>
<td>Face, Eyes, Lmks</td>
<td>2018</td>
<td>CNN+LSTM</td>
<td>EyeDiap</td>
<td>-</td>
<td>-</td>
<td>3.38°</td>
</tr>
<tr>
<td>Gaze360 [117]</td>
<td>Face</td>
<td>2019</td>
<td>CNN+LSTM</td>
<td>Gaze360</td>
<td>-</td>
<td>-</td>
<td>11.1°</td>
</tr>
<tr>
<td>SAGE [82]</td>
<td>Eyes, Lmks</td>
<td>2019</td>
<td>CNN</td>
<td>GazeCapture</td>
<td>2.72 1.78</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TAT [78]</td>
<td>Face, Eyes</td>
<td>2019</td>
<td>CNN</td>
<td>GazeCapture</td>
<td>2.66 1.77</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>RSN [270]</td>
<td>Face</td>
<td>2020</td>
<td>CNN</td>
<td>MPIIGaze</td>
<td>-</td>
<td>-</td>
<td>4.50°</td>
</tr>
<tr>
<td>GoogleGaze [232]</td>
<td>Eyes, Lmks</td>
<td>2020</td>
<td>CNN</td>
<td>GazeCapture</td>
<td>-</td>
<td>1.92</td>
<td>-</td>
</tr>
<tr>
<td>EyeNet [185]</td>
<td>Eyes</td>
<td>2020</td>
<td>CNN+GRU</td>
<td>EVE</td>
<td>3.85 3.48</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>AFF-Net [14]</td>
<td>Face, Eyes</td>
<td>2021</td>
<td>CNN</td>
<td>GazeCapture</td>
<td>2.30 1.62</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>iMon [102]</td>
<td>Face, Eyes</td>
<td>2021</td>
<td>CNN</td>
<td>GazeCapture</td>
<td>1.94 1.49</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GAZEL [184]</td>
<td>Face, Eyes, Lmks</td>
<td>2021</td>
<td>CNN</td>
<td>Private dataset</td>
<td>2.91</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PureGaze [28]</td>
<td>Face</td>
<td>2021</td>
<td>CNN+SA+MLP</td>
<td>ETH-XGaze</td>
<td>-</td>
<td>-</td>
<td>4.50°</td>
</tr>
<tr>
<td>GazeTR [29]</td>
<td>Face</td>
<td>2021</td>
<td>ViT</td>
<td>MPIIFaceGaze</td>
<td>-</td>
<td>-</td>
<td>4.00°</td>
</tr>
<tr>
<td>EVE-SCPT [13]</td>
<td>Eyes</td>
<td>2022</td>
<td>CNN+GRU</td>
<td>EVE</td>
<td>2.75 2.49</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GazeAttentionNet [96]</td>
<td>Face, Eyes, Grid</td>
<td>2022</td>
<td>CNN+MLP</td>
<td>ETH-XGaze</td>
<td>-</td>
<td>-</td>
<td>4.5°</td>
</tr>
<tr>
<td>L2CS-Net [1]</td>
<td>Face</td>
<td>2022</td>
<td>CNN</td>
<td>Gaze360</td>
<td>-</td>
<td>-</td>
<td>9.02°</td>
</tr>
<tr>
<td>GazeCLR [109]</td>
<td>Face</td>
<td>2022</td>
<td>CNN</td>
<td>EVE</td>
<td>-</td>
<td>-</td>
<td>4.15°</td>
</tr>
<tr>
<td>OneEye-Net [5]</td>
<td>Eye</td>
<td>2022</td>
<td>CNN</td>
<td>GazeCapture</td>
<td>2.31</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SE-Gaze [178]</td>
<td>Face</td>
<td>2022</td>
<td>CNN+SE+MLP</td>
<td>MPIIFaceGaze</td>
<td>-</td>
<td>-</td>
<td>4.04°</td>
</tr>
<tr>
<td>FreeGaze [44]</td>
<td>Face</td>
<td>2022</td>
<td>CNN</td>
<td>ETH-XGaze</td>
<td>-</td>
<td>-</td>
<td>2.95°</td>
</tr>
<tr>
<td>HAZE-Net [258]</td>
<td>Face, Eyes, Lmks</td>
<td>2022</td>
<td>CNN</td>
<td>EyeDiap</td>
<td>-</td>
<td>-</td>
<td>4.12°</td>
</tr>
<tr>
<td>GazeNAS [174]</td>
<td>Face</td>
<td>2023</td>
<td>ViT</td>
<td>MPIIFaceGaze</td>
<td>-</td>
<td>-</td>
<td>3.96°</td>
</tr>
</tbody>
</table>

a weakly supervised algorithm for learning 3D gaze information by enforcing scene-level geometric 3D and 2D LAEO constraints between pairs of faces. Dubey et al. [45] propose an unsupervised learning technique based on a large in-the-wild dataset that contains many facial images from the Web. They localise the pupil-centre of each eye and use them to determine the region in which the subject is looking. Similarly, Yu et al. [257] also propose to learn gaze representation from unannotated eye images and then use a few labelled calibration samples for gaze estimation.

4.3.3 Post-processing. Depending on the available data for training gaze estimation algorithms and the requirements of the applications, there is a need to convert between 2D and 3D gaze. The conversion is performed by rotating and translating the camera coordinate system (CCS) and screen coordinate system (SCS) [31]. To convert a 2D gaze point to a 3D gaze direction, we first obtain the rotation \( R_b \) and translation \( T_b \) matrices of SCS with respect to CCS by geometric calibration. With these two matrices, we can compute the 3D gaze target with \( t = R_b [u, v, 0]^T + T_b \), which is the intersection of gaze direction and the screen. The target \( t \) will be used to derive
the 3D gaze direction \( g = k(t - o) \), where \( k \) is the factor \( \frac{1}{|R - o|} \) and \( o \) is the gaze origin; for example, the face or eye centre.

To convert a 3D gaze direction to a 2D point, we will still require \( R_s \) and \( T_s \) and the gaze origin and then revert the calculation process. First, we need to calculate the 3D gaze target vector \( t = (x, y, z) \) which is the interaction of gaze direction to the screen. To do so, we use two equations: the line of sight and the screen plan. Given the origin \( o = (o_x, o_y, o_z) \) and a 3D gaze direction \( g = (g_x, g_y, g_z) \), we can get the equation of the line of sight as

\[
\frac{x - o_x}{g_x} = \frac{y - o_y}{g_y} = \frac{z - o_z}{g_z}.
\]

The equation of the screen plane is the relation between the target vector \( t \) and the rotation matrix \( R_s \) and the translation matrix \( T_s \). From \( R_s \), we can derive a normal vector of screen plane \( n = R_s [1, 2] = (n_x, n_y, n_z) \). Given \( T_s = [t_x, t_y, t_z] \), we can deduce the equation of screen plane as

\[
n_x x + n_y y + n_z z = n_x t_x + n_y t_y + n_z t_z.
\]

Solving the above equations 5 and 6 gives us the target vector \( t \). Then we can compute the corresponding 2D gaze point as \( p = (u, v, 0) = R_s^{-1}(t - T_s) \).

With handheld mobile devices, most of the existing interactive systems use 2D gaze points. The reason is that the holding posture of devices often changes, and the relation between the eye and the screen is not stable, which either leads to inaccurate calculation of the direction or requires constant recalibration of the rotation and translation matrices. Nowadays, research effort is increasingly devoted to 3D gaze direction; for example, Zhang et al. have set up a multi-camera system to capture over 500 gaze directions with various illumination conditions [268]. 3D gaze directions depending on the coordinate system of the head and face may improve robustness for mobile devices in complex environments.

### 4.3.4 Calibration

Eye-tracking systems often need to be calibrated; otherwise, the gaze output might incur too large errors to be useful. Calibration is the process of adjusting and customising the gaze output to reflect the spatial geometry of the camera, the screen, and personal difference [42] in order to improve the estimation accuracy. The calibration process consists of data collection and training.

**Data collection.** The common way to collect ground-truth data for calibration is to design an interactive interface to guide a user’s gaze attention. The point-based method is the most used example, where users are asked to fixate at a point for a few seconds. There are often between 5 and 16 points being displayed consecutively at various locations on a screen [42]. Another popular method is pursuit-based, where users are asked to follow their gaze on a moving target, and the trajectory of the movement can be any shape such as a circle [42] or a rectangle [147]. Compared to the point-based method, the pursuit-based method requires less time to collect the same number of data points and provides a better user experience.

The above data collection is considered as explicit, requiring the users’ attention and voluntary actions. In contrast, implicit calibration collects ground-truth data from the background by estimating user attention in possible fixation locations; for example, mouse and keyboard events [98, 114, 156], typing on the on-screen keyboard and screen touch events [108, 246]. A recent study [256] employs visual saliency, which has distinctive perceptual properties from their surroundings that attract users’ gaze attention [4, 130]. For example, independent continuous objects such as a sailing speedboat and a bright moon in the dark in the video are used as visually salient locations for collecting calibration points implicitly [256].

**Calibrator Training.** Table 2 presents a list of calibration techniques. Earlier calibrators are often adaptive linear regression techniques [159]. Recently, domain adaptation techniques have been applied to tackle the personalised calibration problem. For example, Cui et al. [35] have applied Geodesic Flow Kernel (GFK) to adapt the gaze estimator trained on adult data (the source domain) to predict gaze on children (the target domain).
Table 2. A List of Calibration Methods on Handheld Mobile Devices

<table>
<thead>
<tr>
<th>Project</th>
<th>Year</th>
<th>Data Collection</th>
<th>Calibrator</th>
<th>PoG(cm) Tablet</th>
<th>PoG(cm) Phone</th>
<th>Direction(◦)</th>
</tr>
</thead>
<tbody>
<tr>
<td>iTracker [138]</td>
<td>2016</td>
<td>13-Point</td>
<td>SVR</td>
<td>2.12</td>
<td>1.34</td>
<td>-</td>
</tr>
<tr>
<td>FAZE [186]</td>
<td>2019</td>
<td>-</td>
<td>MAML</td>
<td>-</td>
<td>-</td>
<td>3.08°</td>
</tr>
<tr>
<td>GoogleGaze [232]</td>
<td>2020</td>
<td>Points</td>
<td>SVR</td>
<td>-</td>
<td>0.46</td>
<td>-</td>
</tr>
<tr>
<td>GazeRefineNet [185]</td>
<td>2020</td>
<td>-</td>
<td>Visual Saliency</td>
<td>-</td>
<td>2.75</td>
<td>2.49°</td>
</tr>
<tr>
<td>GazeL [184]</td>
<td>2021</td>
<td>-</td>
<td>Touch Event</td>
<td>SVR</td>
<td>-</td>
<td>1.58</td>
</tr>
<tr>
<td>vGaze [256]</td>
<td>2021</td>
<td>-</td>
<td>Visual Saliency</td>
<td>Clustering + LR</td>
<td>-</td>
<td>1.51</td>
</tr>
<tr>
<td>DAGE [149]</td>
<td>2021</td>
<td>9-Point</td>
<td>MLP</td>
<td>2.43</td>
<td>1.58</td>
<td>-</td>
</tr>
<tr>
<td>iMon [102]</td>
<td>2021</td>
<td>5-Point</td>
<td>Kappa angle+LR</td>
<td>-</td>
<td>1.59</td>
<td>1.11</td>
</tr>
<tr>
<td>EasyGaze [27]</td>
<td>2022</td>
<td>9-Point</td>
<td>LR</td>
<td>-</td>
<td>-</td>
<td>1.93°</td>
</tr>
<tr>
<td>DynamicRead [147]</td>
<td>2023</td>
<td>Pursuit</td>
<td>SVR</td>
<td>-</td>
<td>0.95</td>
<td>-</td>
</tr>
</tbody>
</table>

Note: SVR - Support Vector Regression; MAML - Model-Agnostic Meta-Learning; MLP - Multi Layer Perceptron; LR - Linear Regression without a specific model; * - GazeRefineNet is a label-free PoG refinement model that employed visual saliency.

With deep learning, transfer learning has been widely attempted, that is, either fine-tuning the fully connected layer from a pre-trained CNN model [265] or taking the features extracted on a CNN to train a Support Vector Regression (SVR) [138, 147, 151, 232]. These techniques are sensitive to environmental changes [138, 159]. To overcome this problem, Wang et al. [242] introduce a Bayesian adversarial learning technique in which an adversarial learning block is employed to learn generalisable gaze features to various appearances and head poses.

The FAZE project [186] applies a few-shot learning technique, Model-Agnostic Meta-Learning (MAML), to tailor a gaze estimation model to an individual only with a few calibration samples. It first learns robust features via an encoder-decoder architecture, which captures latent features on head orientation, gaze direction, and the appearance around the eye regions. Then it fine-tunes the model with a few calibration samples on individuals. Often the size of these samples is small and thus leads to an overfitting problem. To resolve this problem, MAML is adopted, which employs 2-step gradient updates to learn the optimal weights on the person-specific models. It has the advantage of minimising the generalisation loss of the network [186].

To calibrate the estimation model for individual users, there are efforts from the interaction perspective; that is, using the relevant eye movements generated during the interaction for calibration [114] and accuracy maintenance [97]. We will discuss this practice in Section 8.

4.4 Summary

From the analysis in Section 4.2, we observe that mobile devices are increasingly integrating advanced cameras, such as RGB-D and IR cameras, facilitating the development of appearance-based gaze estimation on mobile devices. As shown in Table 1, numerous deep learning techniques have been employed to enhance gaze estimation precision, with the input for gaze estimation evolving from facial landmarks to eye landmarks, and eventually to original face and eye images. The evolution is driven by the challenge of changing holding posture of devices and rotation of heads. For example, full-face images have been used to detect head orientation, eye position, eyelid openness, and eyebrow movement. Such information can be used in conjunction with features extracted from face images. In addition, gaze estimation is highly correlated with eye appearance. Subtle muscle changes in the eye area can lead to a change in gaze direction. Intuitively, the up-and-down movement of the eye drives the
interplay of surrounding muscles, such as the eyelids and iris. Therefore, eye appearance features from the whole frame are increasingly employed in gaze estimation.

Most of deep learning models are built on state-of-the-art CNNs or custom CNNs, while recurrent models and ViT have not demonstrated significant improvements over CNNs. As detailed in Tables 1 and 2, the accuracy of gaze estimation can reach 1.49 cm on mobile phones, this level can support coarse-grained gaze interactions on handheld mobile devices, considering that widget sizes typically range from 0.9 cm to 1.2 cm. With calibration, the precision can be further improved; for example, Lei et al. have applied SVR for pursuit calibration and reduced their errors to 0.95 cm, which enables various types of gaze interfaces to support scrolling actions in a reading application [147]. Much of the current research prioritizes enhancing gaze estimation accuracy in static conditions and fixed postures. Future research should focus on achieving high-accuracy eye-tracking in dynamic real-world scenarios, where users interact with their devices in various natural postures [147, 175].

5 DATASETS

Datasets are an important research aspect in gaze estimation, and we have listed and reviewed all the publicly available gaze datasets in Table 3. Most of these datasets are collected on the RGB cameras alone (17 out of 24), 3 of them are on RGB-D, and 4 are on IR. There is a trend of moving the platform and collection conditions from more controlled, desktop-based environments towards unconstrained, mobile device-based settings. In the early stage, gaze data is often collected in the laboratory environment and subjects are required to fix their head on a chin rest [70, 268]. Figure 5 has shown the collection settings and samples of the gaze datasets. More recently, research attention is gradually shifting to real-world settings with a variety of distances to screens, head poses, and illumination conditions. For example, MPIIGaze [273] consists of over 213,000 images from 15 people looking at different gaze positions. The dataset is collected over three months during daily laptop usage. GazeCapture [138] is one of the pioneers in collecting gaze data in in-the-wild environments, containing over 1.4 million images from over 1400 users. TEyeD [65] is the currently largest public dataset of eye images, which contains over 20 million real-world eye images with gaze vector, eye movement types, and pupil and eyelid. The data is collected when users wearing head-mounted eye trackers are performing various tasks, including outdoor sports, daily indoor activities, and car riding.

We have included the datasets collected on both desktop and mobile devices, even though our focus is on mobile device-based gaze estimation. The reason is that the desktop-based datasets can be used to train a deep learning model for extracting face and eye features. In addition, other benchmark datasets can also be used to train a deep-learning model. For example, the pupil and eyelid datasets [66, 68], face image datasets including VGG-Face [188], Labeled Faces in the Wild dataset (LFW) [95] and YouTube Faces (YTF) [248], Celebfaces [157], BayesianFace [158] can be used to extract facial features, and thus for gaze estimation [22].

6 GAZE ANALYTICS

To design gaze-based applications, we often need to process gaze and analyse them into high-level eye movement patterns. This is currently under investigated in the handheld gaze estimation research area. To bridge the gap, this section will first introduce eye physiology to provide a background on the meaning and function of eye movements, and then illustrate how to process gaze data and define high-level gaze events and eye movement patterns.

6.1 Understanding of Eye Movement

Eye physiology plays a crucial role in gaze estimation and guides the development of gaze-interactive applications. Grasping the principles of eye movements and their relation to human consciousness levels is essential when designing ergonomic applications. Humans gather information about the external environment through their
eyes, which involves continuous voluntary or involuntary movements, enabling the eyes to obtain a steady and continuous visual stimulus. These eye movements can be categorised into various types, such as fixation and saccades, as illustrated in Figure 6.

1. *Microscopic eye movements* encompass tremor, microsaccade, and drift, which underlie eye movements like fixation, saccade, and smooth pursuit [164]. Tremor is a periodic, wave-like eye movement with a frequency of 90-105Hz and is the smallest eye movement. It has potential for visual perception [2, 164, 165]. Microsaccades are small, fast, jerk-like eye movements that occur during voluntary fixational eye movements [164, 204, 209]. They can serve as a signal for fixation detection algorithms [53] and rotate around the point of fixation with small amplitudes [53]. Drift and tremor typically occur simultaneously and between microsaccades.

2. *Fixation* stabilises the retina on a stationary object of interest to precept and process detailed information from the focused area. This eye movement is essential for tasks that require high visual acuity, such as reading or observing fine details. During fixation, microsaccades occur 1-2 times per second, it help to prevent the fading of the retinal image and maintain visual perception during fixation. [53, 165].
Table 3. A summary of publicly available gaze datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Year</th>
<th>Camera</th>
<th>Gaze</th>
<th>Head Move</th>
<th>IC</th>
<th>Distance</th>
<th>Sub</th>
<th>Resolution</th>
<th>Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>Columbia [222]</td>
<td>2013</td>
<td>RGB</td>
<td>2D, 3D</td>
<td>5</td>
<td>1</td>
<td>200cm</td>
<td>56</td>
<td>5184×3456</td>
<td>5,880</td>
</tr>
<tr>
<td>UT Multiview [228]</td>
<td>2014</td>
<td>RGB</td>
<td>2D, 3D</td>
<td>8</td>
<td>1</td>
<td>60cm</td>
<td>50</td>
<td>1280×1024</td>
<td>64,000</td>
</tr>
<tr>
<td>EyeDiap [70]</td>
<td>2014</td>
<td>RGB-D</td>
<td>2D, 3D</td>
<td>3 + C</td>
<td>2</td>
<td>80-120cm</td>
<td>16</td>
<td>1920×1080</td>
<td>62,500</td>
</tr>
<tr>
<td>OMEG [84]</td>
<td>2015</td>
<td>RGB</td>
<td>3D</td>
<td>C</td>
<td>10</td>
<td>varying</td>
<td>50</td>
<td>1280×1024</td>
<td>44,827</td>
</tr>
<tr>
<td>SynthesEyes [249]</td>
<td>2015</td>
<td>RGB</td>
<td>3D</td>
<td>C</td>
<td>4</td>
<td>varying</td>
<td>10</td>
<td>120×80</td>
<td>11,382</td>
</tr>
<tr>
<td>MPIIGaze [271]</td>
<td>2015</td>
<td>RGB</td>
<td>2D, 3D</td>
<td>D</td>
<td>15</td>
<td>40-60cm</td>
<td>15</td>
<td>1280×720</td>
<td>213,659</td>
</tr>
<tr>
<td>GazeFollow [200]</td>
<td>2015</td>
<td>RGB</td>
<td>3D</td>
<td>C</td>
<td>D</td>
<td>varying</td>
<td>130,339</td>
<td>variable</td>
<td>122,143</td>
</tr>
<tr>
<td>GazeCapture [138]</td>
<td>2016</td>
<td>RGB</td>
<td>2D</td>
<td>C</td>
<td>D</td>
<td>varying</td>
<td>1474</td>
<td>640×480</td>
<td>2,445,504</td>
</tr>
<tr>
<td>UnitEyes [250]</td>
<td>2016</td>
<td>RGB</td>
<td>3D</td>
<td>C</td>
<td>D</td>
<td>0.5-3cm</td>
<td>N/A</td>
<td>400 × 300</td>
<td>1,000,000</td>
</tr>
<tr>
<td>MPIIGazeFace [272]</td>
<td>2017</td>
<td>RGB</td>
<td>2D, 3D</td>
<td>C</td>
<td>D</td>
<td>varying</td>
<td>15</td>
<td>1280×720</td>
<td>37,639</td>
</tr>
<tr>
<td>TabletGaze [101]</td>
<td>2017</td>
<td>RGB</td>
<td>2D</td>
<td>C</td>
<td>1</td>
<td>30-50cm</td>
<td>51</td>
<td>1280×720</td>
<td>1,785</td>
</tr>
<tr>
<td>InvisibleEye [231]</td>
<td>2017</td>
<td>RGB</td>
<td>2D</td>
<td>C</td>
<td>1</td>
<td>0.5-2cm</td>
<td>17</td>
<td>5 × 5</td>
<td>280,000</td>
</tr>
<tr>
<td>RT-GENE [33]</td>
<td>2018</td>
<td>RGB-D</td>
<td>3D</td>
<td>C</td>
<td>1</td>
<td>80-280cm</td>
<td>15</td>
<td>1920×1080</td>
<td>122,531</td>
</tr>
<tr>
<td>Gaze360 [117]</td>
<td>2019</td>
<td>RGB</td>
<td>3D</td>
<td>C</td>
<td>D</td>
<td>varying</td>
<td>238</td>
<td>4096×3382</td>
<td>172,000</td>
</tr>
<tr>
<td>NVGaze [124]</td>
<td>2019</td>
<td>IR</td>
<td>2D</td>
<td>1</td>
<td>1</td>
<td>0.5-3cm</td>
<td>30</td>
<td>1280×960</td>
<td>4,500,000</td>
</tr>
<tr>
<td>SHTechGaze [154]</td>
<td>2018</td>
<td>RGB</td>
<td>2D</td>
<td>D</td>
<td>15</td>
<td>128×128</td>
<td>137</td>
<td>1920×1030</td>
<td>233,796</td>
</tr>
<tr>
<td>EVE [185]</td>
<td>2020</td>
<td>RGB</td>
<td>2D, 3D</td>
<td>C</td>
<td>1</td>
<td>varying</td>
<td>54</td>
<td>1920×1080</td>
<td>12,308,334</td>
</tr>
<tr>
<td>ETH-XGaze [268]</td>
<td>2020</td>
<td>RGB</td>
<td>3D</td>
<td>C</td>
<td>16</td>
<td>100cm</td>
<td>100</td>
<td>6000×4000</td>
<td>1,083,492</td>
</tr>
<tr>
<td>GW [135]</td>
<td>2020</td>
<td>RGB</td>
<td>3D</td>
<td>C</td>
<td>1</td>
<td>0.5-3cm</td>
<td>19</td>
<td>1920×1080</td>
<td>5,800,000</td>
</tr>
<tr>
<td>LAEO [134]</td>
<td>2021</td>
<td>RGB</td>
<td>3D</td>
<td>C</td>
<td>D</td>
<td>varying</td>
<td>485</td>
<td>variable</td>
<td>800,000</td>
</tr>
<tr>
<td>GOO [230]</td>
<td>2021</td>
<td>RGB</td>
<td>3D</td>
<td>C</td>
<td>D</td>
<td>varying</td>
<td>100</td>
<td>variable</td>
<td>201,552</td>
</tr>
<tr>
<td>OpenNEEDS [52]</td>
<td>2021</td>
<td>IR</td>
<td>3D</td>
<td>1</td>
<td>1</td>
<td>0.5-3cm</td>
<td>54</td>
<td>128×71</td>
<td>2,086,507</td>
</tr>
<tr>
<td>TIEye [65]</td>
<td>2021</td>
<td>IR</td>
<td>2D, 3D</td>
<td>C</td>
<td>1</td>
<td>0.5-3cm</td>
<td>132</td>
<td>variable</td>
<td>20,867,073</td>
</tr>
</tbody>
</table>

Note: The columns of the above table are: (1) the publication year; (2) the type of camera; (3) type of gaze (2D PoG or 3D direction); (4) the number of variations in Head Move in reference to the screen, and C refers to continuous head movements; (5) illumination conditions (IC) where a number in this column refers to the number of illumination conditions being considered, D refers to the ambient light in normal daytime conditions; (6) the distance to the camera(s); (7) the number of Subjects; (8) the resolution of each image; and (9) the number of images.

(3) **Saccades** are rapid, ballistic eye movements that occur between fixations, enabling the eye’s fovea to continuously locate and track new objects in the visual field [47]. As the fastest eye movements, saccades have speeds ranging from 30 to 900 degrees per second, influenced by factors such as target distance, amplitude, and individual differences. Saccades play a crucial role in tasks like reading, environmental scanning, and searching objects of interest [47, 209].

(4) **Smooth Pursuit** is the action of stabilising the gaze on a moving visual target. Smooth pursuit has three phases: initiation, maintenance, and termination [69]. In the initiation phase, a delay of 100-130ms occurs between the target’s movement and the start of smooth pursuit, with the first 100ms of smooth pursuit being in the open-loop phase [21]. It generally reaches peak eye velocity of 30° per second within 220-330ms after the response onset at the target. During maintenance, the eye may exhibit 3-4Hz oscillations for corrective shifts to realign the target image at the fovea. When the target stops, the smooth pursuit movement typically ends within 100ms [69, 203]. The human eye can use numerous available signals to...
provide cues and predictions for future smooth pursuit movements; for example, eyes continue to pursue
the target after it disappears or during occlusion [137]. Intuitively, the velocity of smooth pursuit lies
between that of saccades and fixation.

(5) Blinking protects the eyes by spreading tears to the corneal surface and blinking periodically keeps the
cornea moist. Fixation and blink frequency can be affected by external factors such as humidity and
illumination condition as well as internal factors such as cognitive load and fatigue [164].

In human visual systems, these eye movements are driven by the attention mechanisms: bottom-up and top-
down, which have formed the basis for different gaze interactive applications. The former mechanism refers to the
fact that the fixation point of the eye may be changed by external stimuli that direct attention to discriminative
areas of the scene. The latter mechanism is driven by internal stimuli of cognition, which often reflect users’
intentions. It makes information available in working memory, and people will consciously pay attention to
the scene area that is important to the current behavioural target or task [6, 16]. Sattar et al. [212] learn the
compatibility between user fixation scan path and potential targets to predict the correct target image. Wang
et al. [243] design a model that learns visual saliency and information visualisation of scan paths based on a
sequence of eye movements. These explorations based on gaze and eye movements have great potential for
predictive reasoning about users’ intentions and interests. The visual attention mechanisms have formed the
foundation of these gaze interactions.

6.2 Gaze Data Analytics
Commercial eye-tracking instruments, which generally come with data processing software such as Tobii
Studio, process and transform raw eye-movement data into basic gaze events such as fixation, saccade, and
saccade trajectory, and form scan path representation. These events are often defined by velocity, acceleration,
and amplitude of gaze points and can be inferred via the following algorithms: velocity-threshold fixation
identification (I-VT) [75, 113, 210] and identification by dispersion (I-DT) [129], or spatio-temporal dispersion,
the identification by dispersion and duration thresholds (I-DDT) [139, 163, 210].
I-VT aims to separate fixation and saccade from raw gaze points by calculating point-to-point velocity and acceleration, while I-DT and I-DDT achieve so by spatial dispersion; for example, fixation is identified when the speed of gaze points is low, and these points form in a dense cluster. The task of distinguishing between saccade, stationary and smooth pursuit is called a ternary classification task. Threshold-based algorithms for this task require a combination of several basic algorithms, such as Velocity Velocity Threshold Identification (I-VVT) [132], Velocity Dispersion Threshold Identification (I-VDT) [132] and Velocity Movement Pattern Identification (I-VMP) [131]. These algorithms have worked well to distinguish saccade and fixation for commercial eye trackers in the lab setting, but due to their rigid threshold setting, they can result in a large error in detecting smooth pursuit [280] when there is a variety in pupil sizes and viewing directions. The above algorithms supported in the open-source libraries including PyGaze [36], EyetrackingR [38], PyTrack [73], Pupil [115], GazeParser [223], and GazeR [71]. Currently, a wide range of machine learning-based approaches is proposed to tackle this problem, including Bayesian [211], Random Forest [260], Hidden Markov Models [280], DNN [259], CNN and LSTM [226].

However, for appearance-based gaze estimation on handheld devices, the applications only get the raw gaze points or directions, and there is no off-the-shelf software to process them into high-level patterns. The above algorithms might not be immediately applicable, either. The distance and angle between the head and handheld devices can constantly change, which can compromise the performance of these algorithms. Also, for many desktop environments, the user’s gaze point is always within the screen, while for handheld devices, the cameras may only capture partial or occluded faces, or no faces at all (see Figure 7 adapted from [100, 101, 118, 119]).
Table 4. Eye movement data analysis

<table>
<thead>
<tr>
<th>Pipeline</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Clean</td>
<td>Noise Reduction, De-Nulling, Eliminate Outlier and Blink, Smoothing</td>
</tr>
<tr>
<td>Segmentation for Basic Gaze Events</td>
<td>Fixation, Saccade, Smooth Pursuit; Fixation Statistics, Saccade Velocity, Event Duration</td>
</tr>
<tr>
<td>Data Processing</td>
<td>Removal of Implausible Movements, Merging Intra-Threshold Movements</td>
</tr>
<tr>
<td>Data Visualisation</td>
<td>Segmentation for Further Purposes</td>
</tr>
<tr>
<td></td>
<td>Area of Interest (AOI), Scan Path Representation; Gaze Pattern Modelling: Dwell, Pursuit, Gaze Gesture, etc</td>
</tr>
<tr>
<td>Possible Inferences</td>
<td>Modelling: Top-Down or Bottom-Up, Stochastic Processes;</td>
</tr>
<tr>
<td></td>
<td>Inferences: Personal Attributes, Cognitive Process, Attention &amp; Intention</td>
</tr>
<tr>
<td>Possible Applications</td>
<td>UI Control &amp; Adaption, Medicine &amp; Healthcare, User Security &amp; Privacy, etc.</td>
</tr>
</tbody>
</table>

Therefore, an interesting research direction is to design and develop robust gaze pattern detection algorithms, resistant to fluctuated and erroneous gaze estimations. The analysis and processing of eye movement data need to combine segmentation and organisation at different scales depending on the purpose of the task; for example, from the original gaze path to the gaze event segmentation, and then to higher dimensional segmentation and organisation incorporating physiological and cognitive factors.

6.3 Gaze Data Processing Pipeline

Table 4 presents a general pipeline of processing gaze points from the area of eye tracking community [47, 87]. It starts with data cleansing to improve the data quality, including noise reduction, de-nulling and dealing with outliers. Here, null refers to missing data; for example, an eye-tracking device does not report position coordinates or a participant is not looking at the screen. This is often a necessary step for offline eye-tracking data analysis in desktop environments [89, 244]. Gaze event segmentation refers to detecting gaze events such as fixation or saccade from continuous gaze points using the algorithms in Section 6.2.

The intermediate data processing step is to visualise eye-movement data and remove the data that are beyond a reasonable range and merge data; for example, combining immediately adjacent fixation points. Further segmentation could be used to identify the area of interest (AOI) [90], inferring scan trajectories and heatmaps by saccade path and dwell time, as presented in Figure 6. Such information can serve as advanced features for applications.

6.4 Summary

In the process of investigating gaze analytics and the development of gaze-based applications for handheld devices, we have uncovered several key findings that can guide future research and application design in this area.

Understanding the nuances of eye physiology and human visual attention mechanisms is crucial for the effective design of gaze interactive applications. By comprehending how different types of eye movements, such as saccades, fixations, and smooth pursuits, are related to human consciousness levels and attention mechanisms (bottom-up and top-down), researchers and designers can develop more ergonomic and intuitive gaze-based interaction. This understanding can also facilitate predictive reasoning about users’ intentions and interests, which has a significant potential for creating more personalised and effective gaze interaction.

The development of robust gaze pattern detection algorithms is a critical research direction. Current algorithms are faced with the challenges such as changing distance and angle between the head and the device, partial or
occluded face captures, and fluctuating gaze estimations. These factors can cause instabilities in eye movement data, leading to issues such as reduced frequency, incoherent absence, and other inconsistencies in gaze pattern detection. Therefore, creating algorithms that can withstand these variations and provide accurate gaze event segmentation and organisation is essential for the wider adoption of gaze-based applications on handheld devices.

7 GAZE INTERACTION

This section will review the existing gaze-based interactive applications. We first introduce the categorisation of gaze interactions and then describe applications across different platforms, including handheld mobile devices, desktops, Virtual Reality (VR) and Augmented Reality (AR). These applications help to uncover new opportunities for interactive applications on handheld mobile devices. Figure 8 illustrates the overall design flow of a gaze-engaged interaction from eye movement to final testing and optimisation.

![Fig. 8. The workflow of gaze interaction](image)

7.1 Gaze Interaction

Gaze-only interaction can be broadly classified into two main types: implicit and explicit. Implicit interactions involve the interface adapting to the user’s passive gaze behaviour, while explicit interactions require the user to intentionally move their eyes to provide direct input. Implicit interaction is usually tailored to specific tasks, devices, and environmental characteristics. For instance, a reading application might predict users’ reading speed and automatically turn pages by implicitly analysing users’ gaze trajectories [143, 147], or a display system could alter content rendering based on users’ intentions [247]. Such predictions are often achieved by applying machine learning techniques to estimate gaze behaviours and infer human intentions.

Explicit interaction, on the other hand, relies on users’ voluntary and intentional gaze movements for manipulation. This type of interaction typically employs dwell time, pursuit, gaze gestures, or a combination of these techniques, which will be detailed below.

7.1.1 Dwell-time. Dwell time involves a brief fixating on a target for a period of time [48, 160] in order to differentiate between casual viewing and gaze input. This addresses the Midas touch problem [104] where users unintentionally gaze over the potential target and make selections. This technique has been widely used in gaze-only interactions and is useful for interaction techniques that do not require precise gaze estimates, e.g., gaze typing [160, 172]. The recommended threshold for the dwell time is between 200 and 1500 ms for remote eye-tracking system [160, 172], and the exact threshold often requires trials-and-errors on the threshold for a specific task.

7.1.2 Pursuit. Pursuit refers to smooth pursuit eye movements where the eyes follow a moving object [137, 203, 238]. Pursuit measures the match between users’ eye movement and the object’s movement via Pearson
correlation coefficient [41, 54, 235, 238], or machine learning techniques like CNN [226] and Bayesian [211]. It is often used as a calibration technique [24, 276] as mentioned in Section 4.3.4.

7.1.3 Gaze Gesture. Gaze gesture is a sequence of predefined eye movements (or called strokes) [43, 86, 198, 206]. It is a promising alternative to other gaze interaction techniques, especially when the screen is too small to support other techniques [9]. One advantage of gaze gesture is that it can support a large number of commands using a small number of gesture combinations [43]. However, the use of gaze gestures can introduce complexity, as users may have difficulty recalling complex gestures and initiating them physically [162]. There are machine learning-based methods such as Hierarchical Temporal Memory (HTM) [205, 207] and Graph Neural Networks (GNN) [216] to detect and separate gaze gestures from noisy eye movement signals.

7.2 Types of Gaze Interactions

Gaze-based applications are categorised into three groups based on the interaction and how gaze is acquired: explicit gaze interaction, implicit gaze interaction and multi-model gaze interaction [118, 161]. The two main types of explicit gaze interaction applications are gaze typing and gaze interface control. The former allows users to input text via dwell time or gaze gesture [160, 172, 198, 262], while the latter utilises people’s voluntary eye movements and conscious gaze direction to control or communicate with a computer; for example, a user may perform simple horizontal or vertical eye movements to indicate disagreement or agreement [161].

In implicit gaze interaction, there are three main application scenarios: attentive user interfaces, passive eye monitoring, and gaze-based user modelling. The attentive user interface applies users’ natural eye movements rather than expecting particular gaze behaviours for explicit commands; for example, changing the movie plot based on the viewer’s visual interest [237]. Gaze-based user modelling monitors and analyses the dynamics of gaze behaviour over time to understand users’ behaviour, intention, and cognitive processes [269]. For example, eye movement patterns have been used to recognise human activities such as reading or common office activities [19]. The implicit gaze interaction signals can be used to recognise and speculate humans’ latent behaviours, including measurement of users’ preferences [145], attention [56, 57, 169, 177], interests [150], individual stress [99], emotional states [173], and mental disorders such as schizophrenia and autism spectrum disorder [7, 50, 217]. For example, Deng et al. [37] have analysed eye movement of drivers to predict their fixation and understand their attention allocation on scenes, and Pan et al. [182] have explored the use of eye-tracking technologies to predict drivers’ lane-changing intention. Passive eye monitoring is often for diagnostic applications [46, 220] where people’s visual behaviours are recorded for offline processing [46].

Multi-modal gaze interaction refers to the use of gaze alongside other input modalities, such as voice, touch, or hand gestures. Multi-modal gaze interaction can overcome the limitations of other modalities when they are not accurate or when users have difficulty interacting with them. Gaze can also enhance other modalities; for example, gaze has been combined with touch-based PINs for a more secure authentication solution [121], and gaze is used to improve voice interaction by fixating on an object [167]. Both explicit and implicit gaze interactions can be employed in multi-modal interactions, offering a flexible and intuitive user experience.

In the context of handheld mobile devices, the applications mainly fall into the first two categories, and the trend is moving from gaze-only interaction to multi-model interaction. As listed in Table 5, earlier applications have used explicit gaze gestures to perform particular commands. With the improvement of gaze detection, gaze has served as implicit interaction input; for example, deriving users’ interest or attention by monitoring their gaze in the background [112, 177, 180]. Wrist-worn gaze control [80] provides a gaze-based smart home control setting, which uses off-screen gaze gestures on a wrist-worn unit for IoT control. Gaze+Hold [199] provides a gaze-based interface, where a user can use gaze gesture combined with blink and fixation to complete most of the mouse functionality. For example, the selection of an object is performed by first fixating on an object and...
then closing and opening one eye. The downside of this type of application is that the gaze input might be slow and less accurate than the single-source input, and users may potentially experience fatigue [199].

Also, the applications can be classified as gaze-only and multi-modal interaction [118]. The former uses gaze solely as the interaction input, and this is widely adopted by gaze applications in handheld mobile devices. The latter uses gaze to complement other interaction modalities, such as touching or tilting. Pfeuffer and Gellersen [193] have explored the gaze and touch interaction to extend the area that the thumb cannot reach. For example, a user can look at an element on the screen and tap anywhere, and the system will activate the element.

GTmoPass [120] is an authentication method that uses handheld mobile devices to enter multi-modal passwords; that is, combining gaze gestures (e.g., left to right) with touch input.

7.3 Wider Application Domains

A large number of applications have employed gaze as an interactive modality to complement other modalities in different application domains, including accessibility and productivity, collaboration across devices, device and robot control, and security and privacy.

7.3.1 Accessibility and Productivity. Gaze can be an intuitive interaction modality to be complemented with other modalities to enhance accessibility and productivity. This area of applications can be further grouped into gaming [168], web browsing [142], typing [219], and computer interaction [51, 148, 202, 213]. The main idea is to detect users’ attention and intention from their gaze, and use their gaze dwell time and gesture to trigger different actions. ReType [219] is a gaze-assisted positioning technique, which makes use of users’ gaze to position the
Table 5. Gaze applications on handheld mobile devices

<table>
<thead>
<tr>
<th>Project</th>
<th>Year</th>
<th>Sensor</th>
<th>Feature</th>
<th>Modality</th>
<th>Leverage</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>[40]</td>
<td>2007</td>
<td>Eye Response EBCA</td>
<td>gaze gesture</td>
<td>-</td>
<td>E</td>
<td>interface control</td>
</tr>
<tr>
<td>[48]</td>
<td>2012</td>
<td>external IR cam</td>
<td>dwell &amp; gesture</td>
<td>-</td>
<td>E</td>
<td>target selection</td>
</tr>
<tr>
<td>[145]</td>
<td>2014</td>
<td>Tobii X60</td>
<td>dwell</td>
<td>-</td>
<td>I</td>
<td>attention analysis</td>
</tr>
<tr>
<td>[206]</td>
<td>2015</td>
<td>modified prototype</td>
<td>gaze gesture</td>
<td>-</td>
<td>E</td>
<td>interface control</td>
</tr>
<tr>
<td>[193]</td>
<td>2016</td>
<td>Tobii EyeX</td>
<td>dwell</td>
<td>touch</td>
<td>I</td>
<td>interface control</td>
</tr>
<tr>
<td>[224]</td>
<td>2016</td>
<td>phone camera</td>
<td>gaze basic events</td>
<td>-</td>
<td>E</td>
<td>user authentication</td>
</tr>
<tr>
<td>[125]</td>
<td>2016</td>
<td>FaceLab 5</td>
<td>gaze basic events</td>
<td>-</td>
<td>I</td>
<td>Website usability test</td>
</tr>
<tr>
<td>[150]</td>
<td>2017</td>
<td>phone camera</td>
<td>dwell</td>
<td>-</td>
<td>I</td>
<td>intention inference</td>
</tr>
<tr>
<td>[120]</td>
<td>2017</td>
<td>phone camera</td>
<td>gaze gesture</td>
<td>touch</td>
<td>E</td>
<td>user authentication</td>
</tr>
<tr>
<td>[261]</td>
<td>2017</td>
<td>Tobii eyeX</td>
<td>gaze basic events</td>
<td>touch</td>
<td>I</td>
<td>gaze adaptive UI</td>
</tr>
<tr>
<td>[267]</td>
<td>2017</td>
<td>phone camera</td>
<td>gaze gesture</td>
<td>-</td>
<td>E</td>
<td>gaze input</td>
</tr>
<tr>
<td>[121]</td>
<td>2017</td>
<td>external RGB cam</td>
<td>gaze gesture</td>
<td>touch</td>
<td>E</td>
<td>user authentication</td>
</tr>
<tr>
<td>[227]</td>
<td>2018</td>
<td>phone camera</td>
<td>gaze basic events</td>
<td>-</td>
<td>I</td>
<td>attention inference</td>
</tr>
<tr>
<td>[219]</td>
<td>2019</td>
<td>Tobii 4C</td>
<td>dwell</td>
<td>touch</td>
<td>E</td>
<td>text editing aids</td>
</tr>
<tr>
<td>[202]</td>
<td>2019</td>
<td>Tobii 4C</td>
<td>dwell</td>
<td>touch</td>
<td>E</td>
<td>text interface control</td>
</tr>
<tr>
<td>[167]</td>
<td>2020</td>
<td>phone camera</td>
<td>dwell</td>
<td>voice</td>
<td>E</td>
<td>map navigation</td>
</tr>
<tr>
<td>[220]</td>
<td>2020</td>
<td>phone camera</td>
<td>eye image</td>
<td>-</td>
<td>I</td>
<td>oculom exam</td>
</tr>
<tr>
<td>[230]</td>
<td>2020</td>
<td>phone camera</td>
<td>dwell</td>
<td>touch</td>
<td>E</td>
<td>cross-device control</td>
</tr>
<tr>
<td>[58]</td>
<td>2020</td>
<td>phone camera</td>
<td>dwell</td>
<td>eyelid</td>
<td>E</td>
<td>interface control</td>
</tr>
<tr>
<td>[112]</td>
<td>2020</td>
<td>phone camera</td>
<td>gaze basic events</td>
<td>-</td>
<td>I</td>
<td>attention analysis</td>
</tr>
<tr>
<td>[254]</td>
<td>2021</td>
<td>phone camera</td>
<td>gaze gesture</td>
<td>touch</td>
<td>E</td>
<td>gaze-assist input</td>
</tr>
<tr>
<td>[133]</td>
<td>2021</td>
<td>phone camera</td>
<td>dwell</td>
<td>hand motion</td>
<td>E</td>
<td>interface control</td>
</tr>
<tr>
<td>[180]</td>
<td>2021</td>
<td>Tobii X2</td>
<td>gaze basic events</td>
<td>-</td>
<td>I</td>
<td>attention analysis</td>
</tr>
<tr>
<td>[153]</td>
<td>2021</td>
<td>phone camera</td>
<td>dwell</td>
<td>-</td>
<td>E</td>
<td>target selection</td>
</tr>
<tr>
<td>[123]</td>
<td>2021</td>
<td>Tobii 4C</td>
<td>dwell</td>
<td>voice</td>
<td>I</td>
<td>implicit note-taking</td>
</tr>
<tr>
<td>[122]</td>
<td>2022</td>
<td>external RGB cam</td>
<td>gaze gesture</td>
<td>touch</td>
<td>E</td>
<td>user authentication</td>
</tr>
<tr>
<td>[274]</td>
<td>2022</td>
<td>phone camera</td>
<td>dwell</td>
<td>voice</td>
<td>E</td>
<td>text correction</td>
</tr>
<tr>
<td>[10]</td>
<td>2022</td>
<td>external RGB cam</td>
<td>gaze &amp; face</td>
<td>-</td>
<td>I</td>
<td>user privacy</td>
</tr>
<tr>
<td>[266]</td>
<td>2022</td>
<td>phone camera</td>
<td>eye image</td>
<td>-</td>
<td>I</td>
<td>holding posture detection</td>
</tr>
<tr>
<td>[275]</td>
<td>2022</td>
<td>external RGB cam</td>
<td>gaze basic events</td>
<td>*</td>
<td>E</td>
<td>gaze command definition</td>
</tr>
<tr>
<td>[103]</td>
<td>2022</td>
<td>Tobii X2</td>
<td>gaze basic events</td>
<td>-</td>
<td>I</td>
<td>attention analysis</td>
</tr>
<tr>
<td>[107]</td>
<td>2022</td>
<td>SMI Glasses</td>
<td>gaze basic events</td>
<td>-</td>
<td>I</td>
<td>learning process of typing</td>
</tr>
<tr>
<td>[105]</td>
<td>2022</td>
<td>watch camera</td>
<td>face position</td>
<td>hand motion</td>
<td>E</td>
<td>spatial user interfaces</td>
</tr>
<tr>
<td>[175]</td>
<td>2023</td>
<td>phone camera</td>
<td>dwell, pursuit, gesture</td>
<td>-</td>
<td>E</td>
<td>gaze UI usability test</td>
</tr>
<tr>
<td>[147]</td>
<td>2023</td>
<td>phone camera</td>
<td>dwell, pursuit, gesture</td>
<td>-</td>
<td>I&amp;E</td>
<td>gaze UI usability test</td>
</tr>
</tbody>
</table>

Note: *Feature* means gaze feature for interaction; *Modality* means the other modality with gaze; *Leverage* means the way of leveraging gaze; *E* means Explicit; *I* means Implicit; gaze basic events mean: fixation, saccade, smooth pursuit etc; *:* including eyelids, mouth, and head.

This interaction method reduces the use of the mouse and allows users to perform editing operations while keeping their hands on the keyboard. GazeHelp [148] assists graphic design activities using real-time gaze information. It works as a plugin to Adobe PhotoShop to allow users to select tools with gaze, create windows at the gaze point, and block the current artboard when the gaze is away from the display. It integrates gaze points and a mechanical switch for object selection, positioning, and manipulation. Gaze has been applied to empower disadvantaged people; for example, creating visual design [34] and coding [190].

7.3.2 Collaboration. Prior works also point out that gaze interaction can improve multi-user collaboration efficiency and user experience [85, 144, 192, 214]. Gaze-sharing user interface [214] detects the user’s attention by sensing the user’s eye movements and shares them with other collaborators to enhance team collaboration. GazeChat [85] is a remote communication system that renders gaze-aware 3D profile photos. It uses an off-the-shelf webcam to track users’ gaze in video calls such as Zoom or Teams and then renders gaze to animate the participants’ profile images.

In addition to team collaboration, gaze can also support multi-device interaction. It is increasingly common for a user to have multiple devices or multiple screens. In a collaborative environment with one or more tablets, GazeConduits [239] uses the phone’s front camera to sense and detect the user’s gaze to identify which tablet the

ACM Comput. Surv.
user is looking at and making content selections and actions based on the user’s gaze gestures. GazeMirror [239] allows users to mirror the content between devices by coordinating gaze and four-finger multi-touch gesture. Another gaze+touch project [192] allows multiple users to zoom in and out on a single shared map without interference or occlusion issues.

7.3.3 Control and Interaction. Gaze is playing an important role in controlling devices [110] and robots [152]. Krishna et al. [140] propose a control interface that uses a tablet to process and estimate gaze. The project uses dwell time as a trigger for interaction instruction and manipulates the robot arm to perform a pick-and-drop operation. Jungwirth et al. [110] have designed gaze-triggered actions that use object contours as visual guidance; for example, a user can trace the contour of a lamp to turn on the light.

7.3.4 Security and Privacy. Gaze supports security and privacy in applications. GazeConduits [239] provides user awareness in a poker game; for example, showing playing cards when the gaze looks and hiding them when the gaze leaves. It also maintains the user’s space around the table by detecting who enters and leaves the space at what time and accordingly adjusting the content display. GazeRoomLock [72] combines gaze and head pose for user authentication in VR applications. EyeVeri [224] applies signal processing and pattern matching techniques to explore conscious and unconscious gaze patterns for access authentication. In addition, gaze-based interaction can support authentication and privacy protection on personal devices [116] such as password entry [120] and protection against shoulder surfer [197].

7.3.5 Healthcare and Other Areas. As an important bio-signal, eye movement patterns have been widely used in the diagnosis of mental health such as depression [3], Parkinson’s [81], autism [236], and dyslexia [201]. For example, autism can be diagnosed by analysing joint attention from interactions between eye movement and objects in a room [236]. Gaze is also utilised for assessing the usability of tools or systems [17, 191]. Bace et al. [17] have employed an unsupervised approach to detect gaze contact and attempted to compute and quantify attentional metrics. This allows for analysing how users allocate attention during interactions.

7.4 Summary
We have reviewed a broader scope of applications that make use of gaze as an interaction modality. Most of these applications use commercial eye-tracking devices, including Tobii [51, 148] and Eye tribe [34]. We have seen an increasing number of applications for collaboration, control and interaction, security and authentication developed on mobile devices. In terms of accessibility and productivity, there is a tendency to explore complex actions from gaze, which can be challenging for handheld devices. For example, typing on the phone’s keyboard often requires the gaze to switch back and forth between the target area and the keyboard. This makes it challenging to locate the gaze position accurately and thus can trigger the Midas touch problem. The key to tackling the Midas touch problem is to analyse the eye movement metrics through the cognitive process and separate users’ true intention from the unintentional activities [51, 61, 183, 199]. The other applications that combine fixation and eye-opening/closing actions to select and manipulate objects can be further explored. A potential limitation is that the mobile devices have smaller icons that are presented closely together, which can make the estimation of gaze and inference of gaze trajectories more challenging.

8 FUTURE RESEARCH CHALLENGES AND OPPORTUNITIES
This section focuses on addressing key questions related to the challenges of handheld mobile devices in advancing unconstrained gaze estimation, robust gaze data processing methods and facilitating interaction involving gaze. Our discussion will be centred around three main questions:

- RQ1: How can we achieve robust gaze estimation in unconstrained environments?
RQ2: How can we develop gaze analysis and processing methods that can tolerate the inherent instabilities of dynamic gaze estimation?

RQ3: How can we utilise estimated gaze for a broader range of applications?

8.1 Roadmap for Unconstrained Gaze Estimation

The mobile setting is bringing new challenges to gaze estimation. First of all, the distance and angle between the screen and the eyes might not be ideal or stable. The consequence is that the camera might only capture a partial face or an occluded face; for example, the face could be covered by sunglasses or a scarf (see Figure 7).

Secondly, users tend to change their holding postures, motion states, and whereabouts, which may compromise the gaze estimation and/or calibration model. Thirdly, the variety of environmental conditions, such as lighting, may compromise the quality of images for face and eye detection. In the following, we will present potential directions to tackle these challenges towards achieving unconstrained gaze estimation.

8.1.1 Augmenting with Sensors and Multi-Cameras. One of the major differences between mobile devices and desktop/AR/VR devices is that modern mobile devices are augmented with a rich set of sensors. These sensors provide more information that can reveal the context of the user. For example, we can infer the holding posture from the gyroscope, physical activities from the accelerometer, and the lighting condition from light sensors. Also, we have witnessed an improvement in the camera and sensor technologies on mobile devices. For example, some mobile devices have between three and six cameras with higher resolutions. Combining data from these built-in sensors with the camera may improve the accuracy of gaze estimation [25].

8.1.2 Continuous Calibration. Calibration is often necessary to tune the gaze estimation model and to allow for more accurate estimation adapting to the current user, screen, and environment. Continuous calibration is an interesting direction to explore; for example, quickly adapting the model continuously and obtaining the calibration points without distracting users from their tasks at hand.

A potential direction is to allow for implicit calibration, which leverages users’ interaction and other sensor data in mobile devices, including accelerometer, gyroscope, magnetometer, and compass. Gaze behaviour, such as smooth pursuit, has been explored for calibration. For example, correlating eye or hand movements with the trajectory of moving objects has been utilised for in-use calibration [23, 176]. The trajectory of a straight saccade has been used to calibrate the distortion of eye tracker [97].

Current handheld mobile devices integrate a number of inertial sensors, and their data can be employed for dynamic adjustment. Pino et al. [194] propose to detect changes in the position and holding postures of the phone from gyroscopes and accelerometers. The detected change will inform whether to use current eye images for gaze estimation. Gaze-based input requires a high level of stability in the input process, and device movements and other possible disturbances can decrease the quality of input. A promising direction is to use inertial sensor data to select the input for gaze estimation algorithms or select and tune the gaze estimation algorithms for the current input.

Research can investigate real-time feedback and calibration mechanisms to enhance the resilience of gaze estimation in handheld mobile devices. Real-time feedback on users’ gaze behaviour can be used to calibrate the model on-the-fly, leading to high resilience even in the face of changing conditions.

8.1.3 Diversity in Datasets. As presented in Section 5, current datasets predominantly contain static poses and are limited in terms of capturing gaze estimation in dynamic conditions. There is a need to collect a large amount of data with versatile conditions in more naturalistic settings. However, such data can be expensive to collect; therefore, the current research direction is to explore unsupervised [79], self-supervised [30, 251] and weakly supervised techniques [134].
One increasing concern is bias in face recognition models; for example, the state-of-the-art models are trained on the datasets that over-represent socio-demographic groups with certain skin tones and facial structures, which makes them less accurate for marginalised groups [245]. For example, Buolamwini and Gebru have evaluated three commercial gender classification systems and found that the error rate for classifying lighter-skinned males is 0.8% while the error rate for classifying darker-skinned females can reach up to 34.7% [20]. The bias in the dataset has led to low-quality facial feature extraction on under-represented groups, which can result in low accuracy in gaze estimation. Therefore, future data collection should cover a wider ethnic group of subjects.

8.1.4 Model Deployment. Deploying high-performance gaze estimation models on handheld mobile devices presents challenges in model compression and model deployment.

Model Compression. The existing gaze estimation models, especially the ones based on deep learning, can be computationally expensive and take up much memory; for example, a deep learning model can have billions of parameters. Even though today’s smartphones have much more computational resources, continuously estimating gaze in real-time can still be challenging, and it can compromise their battery life and reduce user experience. Model compression is a popular direction to reduce the size of deep learning models while maintaining model performance. There are several approaches, such as parameter quantisation, parameter pruning, low-rank factorisation and knowledge distillation. Guo et al. [78] apply knowledge distillation and pruning to reduce the size of the CNN model while maintaining its performance.

Model Deployment. Many of the existing gaze estimation models are deployed on the cloud or edge so that mobile devices only perform pre-processing steps and then pass the extracted features to the deep learning models. However, this adds to the communication cost, results in high latency, and increases privacy risk. Future work will look into deploying the models on users’ own devices. However, this raises a practical issue – model framework compatibility.

Most of the deep learning models are implemented in frameworks such as PyTorch or TensorFlow, and these frameworks do not have the same support on different handheld devices. The ONNX (Open Neural Network Exchange) [11] can be used as a medium to transform models across different deep learning frameworks and thus can help deploy models to various platforms of handheld mobile devices. It only supports operators that are commonly supported by all the deep learning frameworks, but not for specialised operators for specific frameworks. This limits the deployment of advanced, customised, gaze estimation models on handheld devices. An engineering perspective of research is how to support gaze estimation on a wide range of different devices.

8.1.5 Open Standards. It is important to highlight the current lack of international or national industrial standards specifically for eye-tracking technologies and systems. Existing standards such as ISO 9241-971:2020 Ergonomics of human system interaction and ISO 13407:1999: Human-centred design processes for interactive systems focus on usability assessment and human-centred design principles. Similarly, WCAG 2.1: Web Content Accessibility Guidelines is for conducting usability studies with people with disabilities, which are relevant to the use of oculomotor systems.

There is a need for the industry to develop appropriate standards that regulate the overall performance and usability of eye-tracking systems. These standards should address various aspects, including device accuracy, calibration procedures, data processing, and user privacy. Establishing such standards could ensure consistent quality and interoperability across different eye-tracking systems, facilitating their adoption and fostering innovation in the field.

8.2 Roadmap for Gaze Analytics

As described in Section 6, there are no established algorithms and toolkits for processing raw gaze points or vectors into high-level gaze patterns or events. Appearance-based gaze estimation on handheld mobile devices
can be much less accurate than commercial eye-tracking devices in terms of large error, low frequency, and high instability. The existing methods of removing outliers, smoothing data, or merging intra-threshold movements might lead to significant information loss, compromising the quality of gaze analytics. Therefore, a future research direction is to develop gaze event detection and processing algorithms to tolerate imperfect gaze estimation. This may require a further understanding of user behaviour patterns and cognitive processes in various applications. Overcoming these challenges will not only facilitate more robust and accurate gaze interaction but also pave the way for a wide range of novel applications in human-computer interaction.

8.3 Roadmap for a Wider adoption of Gaze Applications

8.3.1 Privacy Implication. Gaze interaction can have high privacy implications. First of all, gaze can reveal users’ intentions, interests, emotions, and personality traits [16, 93, 189, 243]. Secondly, gaze estimation from handheld mobile devices is based on users’ facial images. Thirdly, the camera may capture the background, such as bystanders’ activities. Research effort should be devoted to protecting the foreground and background users’ privacy; for example, how to securely store face/eye images, prevent leaking gaze information to third-party applications on the same devices, and explore the possibility of protecting the privacy of bystanders [116].

8.3.2 Gaze Interaction. One obstacle to the wider adoption of gaze interaction on handheld mobile devices is the imprecision of estimation [15] and sensitivity to the environments and interaction positions. Imprecision can increase users’ fatigue and lower user experience. Furthermore, the screen size of handheld mobile devices might limit the choices of gaze pattern combinations; for example, saccade detection on a small screen can be challenging and smooth pursuit or dwell and gesture combination might be a better option.

Promising applications for gaze interaction on handheld mobile devices include healthcare, usability testing, cognitive process understanding, interface and device control, and security and privacy protection. For example, gaze-engaged interface or device control in accessibility contexts can offer an alternative input method for users with motor impairments, and gaze patterns can be used as bio-metric identifiers to enhance security and privacy protection.

Apart from using the front camera to capture the gaze behaviour of the target user, leveraging other cameras, such as rear or external cameras, to capture and project multiple gaze directions and identify objects of interest or perform interaction [117] has rarely been explored. This approach holds promise for opening up a new research area, as it has the potential to link gaze with physical objects or locations in the real world through socialised gaze cues, enabling a more immersive user experience. Gaze-engaged multi-modal interaction is another promising area, as exploring such interactions can complement gaze and other modalities, leading to novel human-computer interaction.

9 CONCLUSION

Mobile human-computer interaction is one of the most popular areas for innovation. This paper has presented a review of gaze estimation and interaction on handheld mobile devices. There are many promising developments in this area: more powerful mobile devices with high-resolution cameras, accurate gaze estimation algorithms based on deep learning, and an increasing number of novel applications that leverage the use of gaze to enable hands-free interaction or complement other interaction modalities. This paper summarises these latest developments and points to the future research challenges and opportunities, especially in gaze estimation accuracy in terms of robustness and continuous calibration, the computational cost in terms of model size and power consumption, and gaze interaction in terms of richer eye movement exploration. It is desirable that the gaze estimation algorithm is adaptable to different environmental conditions and has a low computational overhead. Continuous and implicit calibration will be key factor in supporting unconstrained gaze estimation.

ACM Comput. Surv.
Gaze estimation algorithms are becoming increasingly mature and, although still far from unconstrained estimation, can be applied on handheld mobile devices by installing apps that deploy models. We review the existing types of gaze interactions, summarise various types of eye movements, cognitive theories and combinations of gestures used in gaze interactions, and give a generalised workflow of gaze interactions in the hope that more people will see the role of gaze in interactions and use it to develop more and novel interactions and applications.

REFERENCES


[38] Jacob W Dink and Brock Ferguson. 2015. eyetrackingR: An R library for eye-tracking data analysis.


[182] Yunxian Pan, Qinyu Zhang, Yifan Zhang, Xianliang Ge, Xiaoqing Gao, Shiyan Yang, and Jie Xu. 2022. Lane-change intention prediction from a full face image. In *CVPR ’22*. IEEE, 9992–5000.


34 • Y. Lei et al.


ACM Comput. Surv.


Xiaomeng Zhang, Michael Xuelin Huang, Yusuke Sugano, and Andreas Bulling. 2018. Training Person-Specific Gaze Estimators from User Interactions with Multiple Devices. ACM, 1–12.


