A HIGH-FRAME-RATE EYE-TRACKING FRAMEWORK FOR MOBILE DEVICES

Yuhu Chang[*] , *Changyang He*[*], *Yingying Zhao, Tun Lu*[†], *Ning Gu*

School of Computer Science, Fudan University
Shanghai Key Laboratory of Data Science, Fudan University

ABSTRACT

Gaze-on-screen tracking, an appearance-based eye-tracking task, has drawn significant interest in recent years. While learning-based high-precision eye-tracking methods have been designed in the past, the complex pre-training and high computation in neural network-based deep models restrict their applicability in mobile devices. Moreover, as the display frame rate of mobile devices has steadily increased to 120 fps, high-frame-rate eye tracking becomes increasingly challenging. In this work, we tackle the tracking efficiency challenge and introduce GazeHFR, a biologic-inspired eye-tracking model specialized for mobile devices, offering both high accuracy and efficiency. Specifically, GazeHFR classifies the eye movement into two distinct phases, i.e., saccade and smooth pursuit, and leverages inter-frame motion information combined with lightweight learning models tailored to each movement phase to deliver high-efficient eye tracking without affecting accuracy. Compared to prior art, GazeHFR achieves approximately 7x speedup and 15% accuracy improvement on mobile devices.

Index Terms— gaze estimation, mobile imaging, biomedical video analysis, applications of machine learning

1. INTRODUCTION

Eye-tracking has been an active research topic in the field of human-computer interaction, with potential applications across a wide range of psychological [1, 2] and medical domains, e.g., people with motor impairments [3]. As mobile devices have become the de facto human-computer interface, recent work has been increasingly focusing on developing gaze-on-screen tracking technologies tackling mobile-specific application scenarios [4].

Early solutions for gaze-on-screen tracking suffer from low tracking accuracy due to inherent technology limitations and the lack of sufficient training data [5, 6]. Recently introduced large-scale data sets, e.g., GazeCapture [7], have greatly enhanced the accuracy of learning-based gaze-on-screen models [8, 9, 10, 11], which capture human face related features, determine the relative mapping between the face and screen coordinate systems, and infer human gaze fixation position on the screen. However, adopting learning-based methods to mobile devices still faces serious performance challenges. Specifically, learning-based high-precision eye-tracking models require complex video analytic processing with high computation cost. For instance, ResNet and DenseNet models may consist of over one thousand layers. Using these models, existing methods may require hundreds of milliseconds of CPU time to perform eye tracking at each individual video frame, making it challenging to deliver real-time high-precision eye tracking in high-frame-rate mobile scenarios.

In this work, we tackle the performance challenge by introducing GazeHFR, a highly efficient high-frame-rate eye-tracking method suitable for performance-constrained mobile environment. The proposed work is inspired by the key observation that eye movement can be classified into two distinct phases: *Saccade*, abrupt and rapid movement of both eyes across multiple fixation points, and *smooth pursuit*, slow and smooth eye movement [12, 13]. In contrast to saccade, which is abrupt and rather unpredictable, smooth pursuit is slow and smooth with a strong traceable or predictable motion pattern, which motivates us to develop inter-frame motion prediction technique to improve the efficiency of eye-tracking of smooth pursuit. Furthermore, as smooth pursuit accounts for the majority of eye movements [14], reducing the redundant computation at this stage will bring substantial benefit to the overall eye tracking task performance. To this end, this work presents a new biologic-inspired framework leveraging the characteristics of two eye movement phases. Specifically, GazeHFR performs real-time eye tracking via a two step process. It first detects eye movement in real-time and classifies the movement into saccade or smooth pursuit phase. During a saccade phase, a MobileNet based model is adopted to perform frame-by-frame eye tracking to guarantee tracking accuracy. For smooth pursuit, the most common case, GazeHFR leverages inter-frame motion information to perform highly efficient eye tracking without affecting (or even enhance) tracking accuracy.

In summary, this work makes the following contributions. (1) We propose and develop GazeHFR, the first biologic-inspired learning-based high-frame-rate gaze-on-screen tracking framework. The proposed work targets resource-constrained mobile platforms, with the considera-

[*] indicates an equal contribution. [†] indicates the contact author.

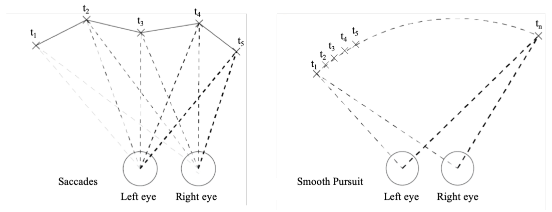

Fig. 1. Saccade (left) and smooth pursuit (right).

tion of motion dynamics and jitters caused by human motions. (2) GazeHFR exhibits superb efficiency on mobile platforms. Compared to prior work [7], GazeHFR offers approximately 7x speedup with 15% accuracy improvement.

2. PROPOSED MODEL

Eye movement patterns can be classified into two distinct phases: *saccade* and *smooth pursuit*. As shown in Fig. 1, Saccade refers to rapid conjugate eye movements that shift both eyes from one target to another. Considered as one of the fastest movements produced by human body, the speed of the saccade motion can be up to 900 degrees/s [12]. Smooth pursuit refers to slow and smooth eye movements with minimized retinal target motion with speed mostly lower than 30 degrees/s. The two phases typically occur in an alternating fashion [15], and smooth pursuit accounts for the majority time amount [14]. Such eye movement characteristics motivate us to design techniques to detect, classify and track each eye movement phase both accurately and efficiently.

2.1. System Overview

As shown in Fig. 2 and Fig. 3, GazeHFR consists of three key components, a classification module, a single-frame module (SFM) for saccade, and a time-series module (TSM) for smooth pursuit. The classification module aims to detect and classify which phase of real-time eye movement belongs to, i.e., saccade or smooth pursuit. If a saccade phase is detected, the single-frame module is triggered, which uses a MobileNet-based deep model to perform frame-by-frame high-precision eye tracking to guarantee tracking accuracy. Otherwise if a smooth pursuit phase is detected, the time-series module is triggered, and inter-frame motion information is leveraged to perform highly efficient eye movement prediction, and the costly frame-by-frame high-precision tracking is avoided.

2.2. Classification Module

The classification model uses seven carefully selected features to accurately characterize the eye-device relative movement (Fig. 4), specifically, $(\Delta x, \Delta y)$ of pupil-to-corner displacement of each eye (2 dimensions $\times 2$ eyes) and ($\Delta Pitch$, $\Delta Roll$, ΔYaw) of devices. We detect the eye corners and

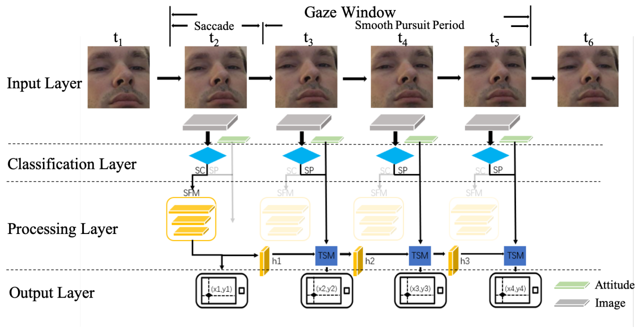

Fig. 2. Flowchart of the proposed GazeHFR.

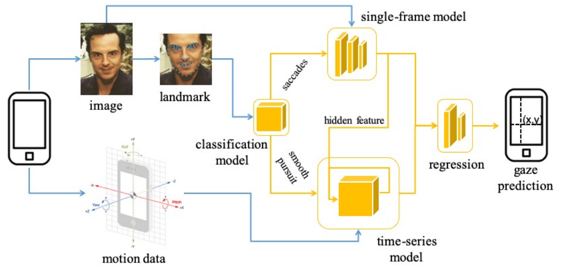

Fig. 3. The architecture of GazeHFR.

pupil using Face++ API, a mature facial landmark detection method. To eliminate the impact of face size and rotation in the image, a specialized normalization strategy is designed for the eyeball movement. Without losing generality, we take the right eye as an illustration example. First, $\vec{b}$, the vector from left eye corner to right eye corner is taken as the x-axis, and $|b|$ is defined as per unit length. Further, $\vec{a}$, the vector from the left corner to the pupil is used to calculate the eyeball movement with the formula shown in Fig. 4. In this reference system, we effectively avoid the influence of the inclination and zoom of the face in different real-world settings.

2.3. Single-Frame Module for Saccade

For a saccade phase, the fixation point cannot be accurately predicted solely based on the motion information of prior

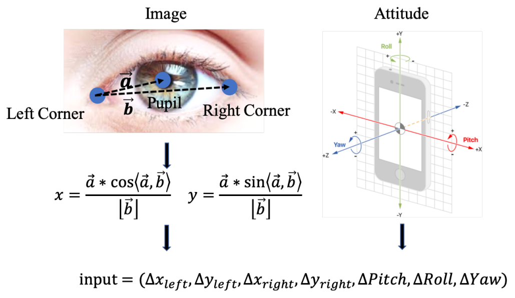

Image
Attitude
Left Corner
Pupil
Right Corner
$\vec{a}$
$\vec{b}$
$x = \frac{\vec{a} * \cos\langle\vec{a},\vec{b}\rangle}{|\vec{b}|}$
$y = \frac{\vec{a} * \sin\langle\vec{a},\vec{b}\rangle}{|\vec{b}|}$
Roll
Yaw
Pitch
input = $(\Delta x_{left}, \Delta y_{left}, \Delta x_{right}, \Delta y_{right}, \Delta Pitch, \Delta Roll, \Delta Yaw)$

Fig. 4. Eye classification features.

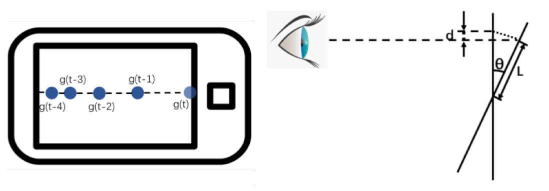

Fig. 5. Translational and rotation motion.

frames. Therefore, SFM requires the input from the full-scale image frame captured by the mobile device. In GazeHFR, we adopt MobileNetV2 [16] as the backbone of SFM. MobileNetV2 is widely used as a convolutional neural network model for mobile devices since it is lightweight and efficient. In addition, MobileNetV2 can also deliver acceptable performance for location detection tasks.

2.4. Time-Series Module for Smooth Pursuit

For a smooth pursuit phase, there are three possible cases: (1) **translational**: gaze movement caused by translational motion of eyes or mobile device, e.g., reading a line of text; (2) **rotation**: gaze movement caused by the rotation of a mobile device, e.g., device shake; and (3) **combination**: gaze movement combining translational motion and rotation. The gaze motion trace may vary with different trajectory shapes (anomaly, arc or approximately linear shape). As the third case is the combination of the first two, the following discussion only needs to cover the first two cases.

Translational: Without loss of generality, we discuss uniform-acceleration linear gaze motion to simulate the instantaneous gaze movement. Fig. 5 (left) exhibits the linear gaze motion with uniform acceleration. Since the acceleration remains constant, the trace can be formulated as:

$$\begin{aligned} y &= y_{g(t-1)} + \Delta y_{g(t-1)} + (\Delta y_{g(t-1)} - \Delta y_{g(t-2)}) \\ x &= x_{g(t-1)} + \Delta x_{g(t-1)} + (\Delta x_{g(t-1)} - \Delta x_{g(t-2)}) \end{aligned} \tag{1}$$

GRU is adopted to capture the inter-frame motion information for translational trace tracking [17], in which the hidden state h^{t-1} of GRU is used to carry the gaze motion features of previous frames. We assume h^{t-1} is a linear representation of $\Delta y_{g(t-1)}, \Delta y_{g(t-2)}, \Delta y_{g(t-3)}, \Delta x_{g(t-1)}, \Delta x_{g(t-2)}, \Delta x_{g(t-3)}$. With x^t including both $y_{g(t-1)}$ and $x_{g(t-1)}$, W^h matrix can easily learn the linear prediction. Then, with the help of intermediate state of GRU cell, h^t can also be restored as the same linear representation form of h^{t-1}. The relationship between adjacent predictions move forward in this pattern.

Rotation: Fig. 5 (right) illustrates a simulative gaze motion scenario caused by device rotation. when the mobile phone rolls θ degrees to be perpendicular to the line of sight, the gaze point moves equals $L \times (1 - sin\theta)$. Considering complex mobile scenarios where slight rotation-based gaze movements occur frequently (e.g., jitter). Such movement noise often contains three types of motion patterns, i.e., *roll, yaw, and pitch*. To tackle this problem, GazeHFR leverages mobile motion sensor to capture and compensate for such motion noise during eye tracking.

Table 1. Classification Performance of Different Methods.

Model	Seq Length	Precision (%)	Recall (%)	Time (ms)
SVM	5	77.30	77.30	-
RF	5	77.30	86.71	-
XGBoost	2	79.12	**94.99**	**1.34**
	3	80.71	94.30	1.45
	5	82.71	94.59	1.57
GRU	2	98.00%	92.29%	3.20
	3	**98.26%**	**94.21%**	3.19
	5	97.99%	90.52%	3.20

3. EXPERIMENT

3.1. Experimental Setup

Dataset Preparation: We use the GazeCapture dataset [7] to evaluate the performance of GazeHFR. We follow the split of train/validation/test set in GazeCapture. That is, the train/validation/test set contains 1,251,983/59,480/179,496 frames. We discard 32,751 invalid frames with missing timestamps or motion data. We take the period of fixation at specific dot in screen in dataset as smooth pursuit considering the relative motion of head and device.

Compared Methods: GazeHFR is compared against the following methods.

- *iTracker* has shown the best representativeness among existing methods from the literature. The asterisk means the $Error$ is the dot error mentioned in the article for stream of frames [7].
- *Baseline-saccade* is designed as the model in which only the saccade frame is calculated, while the rest is filled with this result in a gaze window. It is to detect whether TSM is working or not.
- *Baseline-single-frame* was applied to every frame on the test set, not only the saccade phase. It is designed to prove that the improvement of efficiency is mainly contributed by GazeHFR rather than MobileNet.

Evaluation metric: $recall$ and $precision$ are used as the performance metric for the classification module. We report the gaze prediction performance using gaze $error$. Here, $error$ is defined as the Euclidean distance between the predicted and the actual gaze position. We also define $efficiency$, which represents the optimized workload of eye tracking per unit calculation time. Here, the optimized workload is defined as the reference model error minus the measured $error$.

Table 2. Comparison of GazeHFR and the iTracker* regarding performance and computation time on the server. iTracker* is selected as the reference model for $efficiency$, so it has no $efficiency$ value.

Model	Time (ms)	$Error$ (cm)	$Efficiency$ (cm/s)
iTracker*	84.15	3.16	-
GazeHFR	**9.58**	**2.68**	**50.10**
Baseline-saccade	**8.87**	2.78	42.84
Baseline-single-frame	38.21	**2.64**	13.61

$Efficiency$ represents the performance of models in terms of computation time and gaze $error$. The less time or the smaller $error$, the higher $efficiency$ and the better.

Implementation: We implemented our model in PyTorch. Experiments were run on a Dell PowerEdge T640 with two Intel Xeon Silver 4110 2.10GHz 8-core processors, 64 GB RAM, and four NVIDIA TITAN V GPUs. We also evaluate our model on a set of mobile devices, i.e., iPhone X, iPhone Xs Max, iPad Pro 11, Pixel 2, and Pixel 3. In particular, we take Apple's exclusive Core ML platform on iOS.

Training SFM: For SFM, we use MobileNetV2 as the backbone to extract features since MobileNetV2 offer excellent efficiency and efficacy tradeoff on mobile devices. In our gaze prediction task, we replace the last layer of MobileNetV2 with two newly-added fully connection (FC) layers: FC1 and FC2. The out channels are squeezed gradually from FC1 to FC2 layers, which are 128 and 2, respectively. The two outputs reflect the two predicted gaze position coordinates: $\hat{x}$ and $\hat{y}$. First, SFM loads the parameters pre-trained on ImageNet dataset [18] except the last regression part. Then, we tune the SFM based on GazeCapture dataset [7]. We choose Adam [19] as the optimizer. The learning rate is initially set as 0.1 and it decays at a rate of 1/10 for each epoch.

3.2. Overall Performance

Eye Movement Classification: Table 1 evaluates the classification performance of the proposed method using the following widely used classifiers: SVM, Random Forest(RF), XGBoost [20], and GRU [17]. For each classifier, the sequence length is set as 2, 3, and 5 respectively. As shown in Table 1, the XGBoost classifier with sequence length 2 achieves the best recall (94.99%) and processing time (1.34 ms). The GRU classifier delivers the best precision (98.26%). The classifiers exhibit acceptable performance, and can be used for the following gaze prediction in GazeHFR.

Gaze Prediction on Server: Table 2 shows the comparison between GazeHFR and iTracker* regarding accuracy and computation time running on the server. The computation time is 9.58 ms (GazeHFR) and 84.15 ms (iTracker*), respectively. The prediction $Error$ is 2.68 cm (GazeHFR) and 3.16 cm (iTracker*). We can see that GazeHFR achieves approximately 9x efficiency improvement and 15% accuracy improvement compared with iTracker*. Interestingly, we find that the baseline-saccade model outperforms iTracker* despite none of the subsequent frames is computed. It shows that the saccade phase of our single-frame model can automatically extract the eyes and location context information, which needs to be input manually in iTracker*.

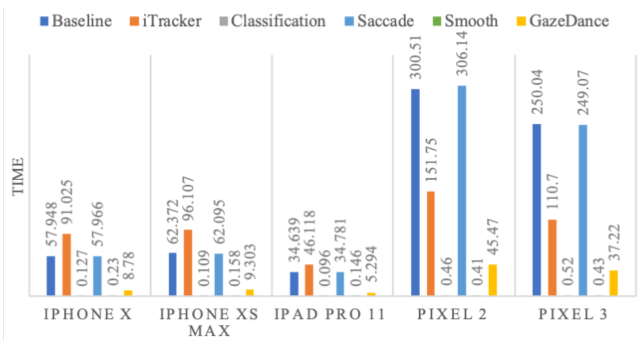

Fig. 6. Comparison under Mobile Scenario.

Compared to baseline-saccade without the time-series module, GazeHFR reduced the error by 0.1 cm, which demonstrates the effectiveness of the time-series model. Compared to baseline-single-frame, GazeHFR loses 1.5% accuracy, but with only 25% of computation time.

Gaze Prediction on Mobile Devices: Fig. 6 visualizes the computation cost of GazeHFR and iTracker* on a set of mobile devices. As we can see, GazeHFR achieves approximately 7x efficiency improvement compared with other baselines. The performance improvement is the same, 15%, as that of the server case since we use the same GazeHFR on both the server and the mobile devices, as GazeHFR can be seamlessly adopted from server to mobile devices.

4. CONCLUSION

In this work, we introduce GazeHFR, a highly efficient high-precision gaze-on-screen tracking framework targeting high-frame-rate yet resource-constrained mobile scenarios. We develop a biologic-inspired and interleaving policy to support eye movement phase-specific tracking. The proposed work leverages efficient motion estimation combined with lightweight deep models to enable efficient yet accurate eye tracking. Experimental results demonstrate that GazeHFR achieves approximately 7x efficiency improvement and 15% accuracy improvement compared with the existing method.

5. ACKNOWLEDGEMENTS

This work was supported by the the National Key Research and Development Program of China under grant No.2016YFB1001404, and National Natural Science Foundation of China (NSFC) under the Grant No. 61932007.

6. REFERENCES

[1] Jung-yueh Tu and Yu-Fu Chien, "The processing of mandarin chinese tonal alternations in contexts: An eye-tracking study," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6244–6248.

[2] Zeeshan Akhtar and Tanaya Guha, "Computational analysis of gaze behavior in autism during interaction with virtual agents," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 1075–1079.

[3] Tomasz Kocejko, Adam Bujnowski, and Jerzy Wtorek, "Eye-mouse for disabled," in *Human-computer systems interaction*, pp. 109–122. Springer, 2009.

[4] Qiong Huang, Ashok Veeraraghavan, and Ashutosh Sabharwal, "Tabletgaze: dataset and analysis for unconstrained appearance-based gaze estimation in mobile tablets," *Machine Vision and Applications*, vol. 28, no. 5-6, pp. 445–461, 2017.

[5] Dongheng Li, David Winfield, and Derrick J Parkhurst, "Starburst: A hybrid algorithm for video-based eye tracking combining feature-based and model-based approaches," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)-Workshops*. IEEE, 2005, pp. 79–79.

[6] Sumit Jha and Carlos Busso, "Estimation of gaze region using two dimensional probabilistic maps constructed using convolutional neural networks," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3792–3796.

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

[8] Braiden Brousseau, Jonathan Rose, and Moshe Eizenman, "Smarteye: An accurate infrared eye tracking system for smartphones," in *2018 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*. IEEE, 2018, pp. 951–959.

[9] Adria Recasens, Petr Kellnhofer, Simon Stent, Wojciech Matusik, and Antonio Torralba, "Learning to zoom: a saliency-based sampling layer for neural networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 51–66.

[10] En Teng Wong, Seanglidet Yean, Qingyao Hu, Bu Sung Lee, Jigang Liu, and Rajan Deepu, "Gaze estimation using residual neural network," in *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, 2019, pp. 411–414.

[11] Kang Wang, Rui Zhao, Hui Su, and Qiang Ji, "Generalizing eye tracking with bayesian adversarial learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11907–11916.

[12] AF Fuchs, "Saccadic and smooth pursuit eye movements in the monkey," *The Journal of Physiology*, vol. 191, no. 3, pp. 609–631, 1967.

[13] Martha G MacAvoy, Jacqueline P Gottlieb, and Charles J Bruce, "Smooth-pursuit eye movement representation in the primate frontal eye field," *Cerebral Cortex*, vol. 1, no. 1, pp. 95–102, 1991.

[14] Valldeflors Vinuela-Navarro, Jonathan T Erichsen, Cathy Williams, and J Margaret Woodhouse, "Quantitative characterization of smooth pursuit eye movements in school-age children using a child-friendly setup," *Translational vision science & technology*, vol. 8, no. 5, pp. 8–8, 2019.

[15] J Kevin O'Regan, "The control of saccade size and fixation duration in reading: The limits of linguistic control," *Perception & Psychophysics*, vol. 28, no. 2, pp. 112–117, 1980.

[16] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[17] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

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

[19] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[20] Tianqi Chen and Carlos Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.