IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received 24 April 2023, accepted 27 June 2023, date of publication 5 July 2023, date of current version 10 July 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3292411

RESEARCH ARTICLE

Predicting Future Eye Gaze Using Inertial Sensors

ARDIANTO SATRIAWAN[1], AIRLANGGA ADI HERMAWAN[1], YAKUB FAHIM LUCKYARNO[1], AND JI-HOON YUN[1,2], (Senior Member, IEEE)

[1]Department of Electrical and Information Engineering, Seoul National University of Science and Technology, Seoul 01811, South Korea
[2]Research Center for Electrical and Information Technology, Seoul National University of Science and Technology, Seoul 01811, South Korea

Corresponding author: Ji-Hoon Yun (jhyun@seoultech.ac.kr)

This work was supported in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) Grant funded by the Korean Government through the Ministry of Science and ICT (MSIT) (Biosignal-based predictive content creation for cloud VR) under Grant 2017-0-00650, and in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant NRF-2019R1A6A1A03032119.

ABSTRACT Eye tracking is a technology that is in high demand, especially for next-generation virtual reality (VR), because it enables foveated rendering, which significantly reduces computational costs by rendering only the area at which a user is gazing at a high resolution and the rest at a lower resolution. However, the conventional eye-tracking technique requires per-eye camera hardware attached near the eyes within a VR headset. Moreover, the detected eye gaze follows the actual eye gaze with a finite delay because of the camera latency, the need for image processing, and the VR system's native latency. This paper proposes an eye-tracking solution that predicts a user's future eye gaze using only the inertial sensors that are already built into VR headsets for head tracking. To this end, we formulate three time-series regression problems to predict (1) the current eye gaze using past head orientation data, (2) the future eye gaze using past head orientation and eye gaze data, and (3) the future eye gaze using past head orientation data only. We solve the first and second problems using machine learning models and develop two solutions for the final problem: two-stage and single-stage approaches. The two-stage approach for the final problem relies on two machine learning models connected in series, one for the first problem and the other for the second problem. The single-stage approach uses a single model to predict the future eye gaze directly from past head orientation data. We evaluate the proposed solutions based on real eye-tracking traces captured from a VR headset for multiple test players, considering various combinations of machine learning models. The experimental results show that the proposed solutions for the final problem reduce the error for a center-fixed gaze by up to 50% and 20% for anticipation times of 50 and 150 ms, respectively.

INDEX TERMS Eye tracking, gaze prediction, virtual reality.

I. INTRODUCTION

Eye tracking technology tracks the point of gaze or the position of the pupil of each eye [1]. It has mostly been considered as the basis of input devices for human–computer interaction. At present, it has become a technology that is in high demand for next-generation virtual reality (VR) systems in order to realize *foveated rendering* to reduce computational overhead [2]. For high-quality VR services, realizing an ideal image resolution (e.g., 4K for each eye) at an ideal frame rate (e.g., 120 frames per second) is essential, but this comes at the expense of significant computing power. Foveated rendering refers to applying a high resolution only in the area that the user is paying attention to and a lower resolution for the rest of the display, which is less perceivable due to the nature of the human visual system [2]. However, to enable foveated rendering, a VR system needs to be able to identify the user's eye gaze in real time.

The conventional implementation of eye tracking for VR is a video-based system using dual cameras attached near the eyes within the headset to detect eye movements via real-time analysis of corneal reflection images [3]. This need for additional hardware increases the cost of the VR headset and limits its form factor. Moreover, due to the latency of the

The associate editor coordinating the review of this manuscript and approving it for publication was Tai-Hoon Kim.

TABLE 1. Specifications of the target problems.

Problem index	Input: Past head orientation	Input: Past eye gaze	Prediction output
1	✔	✘	Current eye gaze
2	✔	✔	Future eye gaze
3	✔	✘	Future eye gaze

cameras, the image processing procedure, and the VR system itself, the detected eye gaze follows the actual eye gaze with a finite delay. The cost increase and the latency problem are both significant, especially for standalone VR headsets with limited cost, computing, and battery budgets. If such delayed eye tracking is used for foveated rendering, areas rendered at low resolution may appear in the user's region of interest, and the user's perception and experience may be degraded. Moreover, in the scenario of VR offloading to a cloud/edge computing entity [4], [5], the conventional approach to eye tracking may make the latency problem more severe while also giving rise to a network bandwidth problem since the dual eye images must be continuously sent at a high frame rate over a wireless connection.

Recent attempts at eye gaze prediction have been made using various alternative methods. In some approaches, the conventional per-eye camera system is replaced with cheaper hardware, such as infrared LEDs paired with photodiodes [6], [7], [8], [9], [10], [11], a smartphone camera [12], [13], [14], [15], or ultrasound sensors [16]. There have also been some attempts to implement eye gaze prediction with no extra hardware using a mathematical model [17], image processing [18] or machine learning (ML) [19], [20]. However, all of the existing works have addressed only the prediction of the user's current eye gaze; no attempt has been made to predict the future eye gaze without extra hardware. Therefore, the existing approaches to eye gaze prediction are unable to solve the abovementioned latency problem faced by eye tracking in various VR systems.

In this paper, we develop a predictive eye-tracking solution that predicts a user's future eye gaze using inertial sensors only, with no need for additional hardware dedicated to eye tracking. Therefore, it is applicable in current VR headsets at no extra hardware cost. Moreover, its ability to predict the user's future eye gaze offsets the VR system's latency and enables responsive eye tracking with a possible additional latency budget. To achieve the prediction task, we formulate three prediction problems, as summarized in Table 1:

- *Problem 1: Predict the current eye gaze from the past head orientation.*
- *Problem 2: Predict the future eye gaze from the past eye gaze and head orientation.*
- *Problem 3: Predict the future eye gaze from the past head orientation.*

Problem 3 is our ultimate goal. That is, when prediction is conducted at time t_0, the goal is to predict the eye gaze at times $\geq t_0$ using the available sensor data extracted at times $\leq t_0$.

First, we observe the relationship between eye gaze and head orientation for different latency cases. These observations show strong correlations, implying that the eye gaze can be predicted from the head orientation. However, our observations also reveal that as the latency increases, a single head orientation sample will become insufficient to predict the eye gaze; instead, the motion path (i.e., time-series data) of the head orientation becomes necessary for prediction.

Then, we solve the first and second problems using various ML models and ultimately develop two solutions for the final (third) problem: a two-stage approach and a single-stage approach. The two-stage approach to the final problem relies on two ML models combined in series, one for the first problem and the other for the second problem. That is, the first-stage model predicts the current eye gaze from the head orientation data, and the second-stage model predicts the future eye gaze from the predicted current eye gaze data. In contrast, the single-stage approach uses a single model to predict the future eye gaze directly from past head orientation data. We evaluate the proposed solutions based on real eye-tracking traces captured from a VR headset for multiple test players, considering various combinations of ML models. The experimental results show that the proposed solutions for the final problem reduce the error for a center-fixed gaze by up to 50% and 20% for anticipation times of 50 and 150 ms, respectively, and that the single-stage approach outperforms the two-stage approach.

The rest of this paper is organized as follows. Recent studies related to eye tracking for VR are reviewed and discussed in Section II. Section III presents our experimental observations on eye movement predictability. We describe the proposed solutions in Section IV and discuss their evaluation results in Section V. Finally, Section VI concludes the paper.

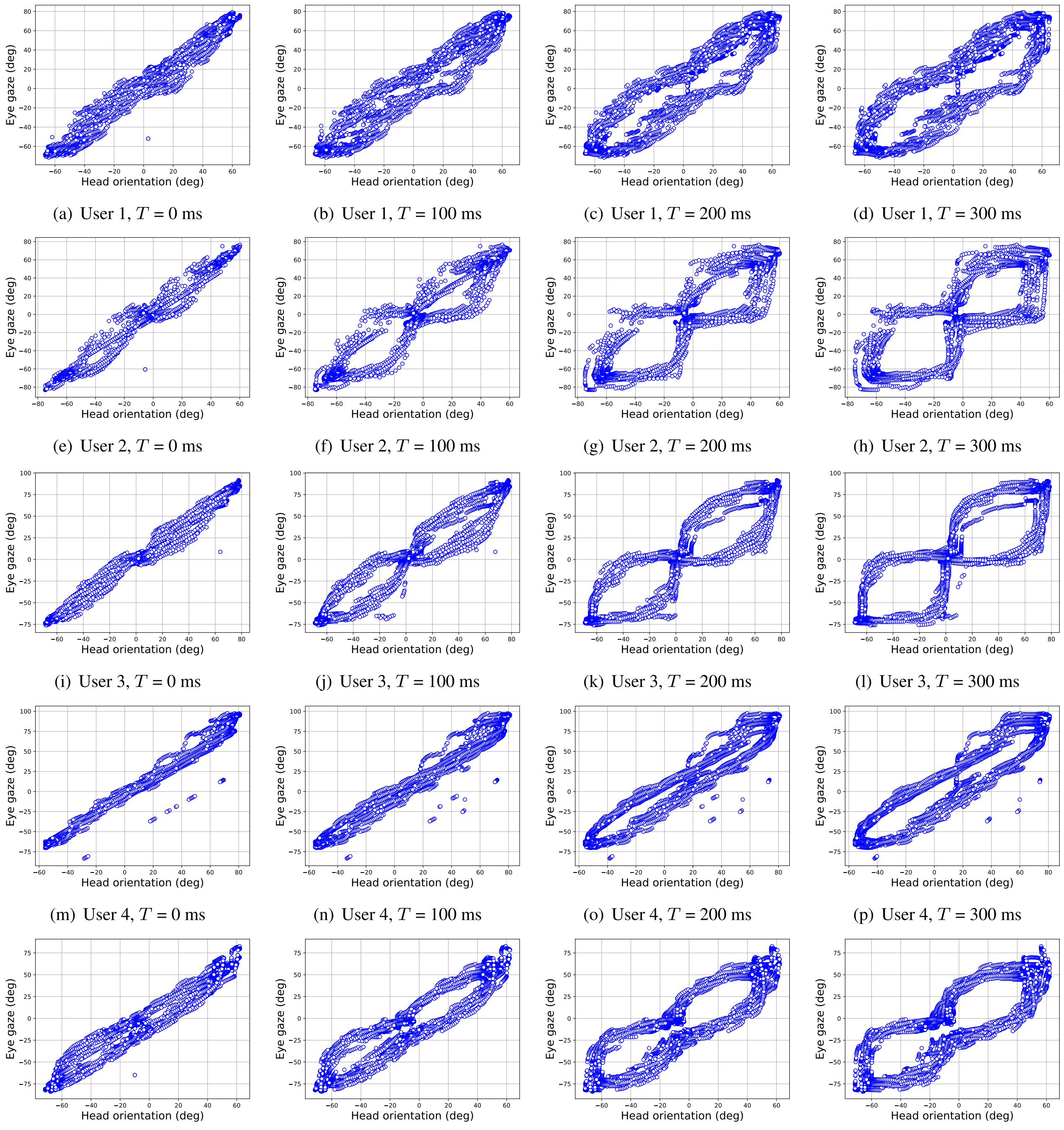
II. RELATED WORKS

The modern method of eye tracking that is most commonly used in current commercial products is video-oculography. In this method, a camera is placed in front of each of the user's eyes and continuously acquires images of the retina. The resulting images are then processed using image-processing techniques to obtain the user's eye position. This technique is expensive because it requires at least two built-in cameras and visual processing of stereo images. To reduce the cost, some researchers have proposed using the front-facing camera of the smartphone in a smartphone-attached VR headset as the tool to acquire the images of the user's eyes. Ahuja et al. [12] and Drakopoulos et al. [13] proposed using the front-facing smartphone camera in such a VR headset to capture the user's

TABLE 2. Summary of related works on eye gaze prediction.

Reference(s)	Objective	Hardware	Method	Input data	Output data
[13], [14]	Low-cost and software-only approach to eye tracking for smartphone-based VR headsets	Smartphone	Image processing and CNN	Image of the user's eyes captured by a smartphone front camera	Current eye gaze
[15]	Eye tracking for general human–computer interaction	Smartphone	Image processing	Image of the user's eye from a smartphone front camera and the captured image of the current VR screen	Current eye gaze
[16]	Eye tracking for smartphone-based VR headsets	Smartphone	Image processing	Image of the user's eye from a smartphone front camera with the lens reflection filtered out	Current eye gaze
[7]	Eye tracking for VR and augmented reality applications with minimal energy consumption	Infrared LEDs and photodiodes	CNN	Amount of reflected infrared light from LEDs captured by an array of photodiodes	Current eye gaze
[8]	Power-efficient eye-tracking devices for VR headsets	Infrared LEDs and cameras	Image processing and CNN	Image of infrared-LED-illuminated eyes captured by infrared cameras	Current eye gaze
[9]	Contact lens device for general eye tracking	Infrared LEDs and photodiodes	Centroid of the currents	Amount of reflected infrared light from LEDs captured by an array of photodiodes embedded in a specialized contact lens	Current eye gaze
[10]–[12]	Eye gaze tracking device with low power/energy consumption	Photodiodes	Neural network	Light from the VR screen reflected by the user's eyes and captured by an array of photodiodes installed in the headset	Current eye gaze
[17]	Eye movement tracking for diagnostics, drug testing, and human–computer interaction	Piezoelectric MEMS transducers	Estimation of the time of flight of acoustic signals	Reflection of ultrasonic acoustic signals captured by piezoelectric MEMS transducer arrays integrated on glasses	Current eye gaze
[22]	Wireless contact lenses for eye tracking in VR applications	Lasers and infrared cameras	Mathematical model	Reflected light from specialized contact lenses with infrared surface-emitting lasers captured by infrared cameras	Current eye gaze
[18]	Estimation of eye gaze using a system dynamics model without an eye-tracking device	Inertial sensors	System dynamics model	Head orientation data captured by the inertial sensors of a wearable device	Current eye gaze
[19]	Finding the area of focus of the eyes from VR screen capture	Inertial sensors	Image processing and ML	Current image from the VR screen and horizontal angular velocity of the head from the inertial sensors of the VR headset	Current eye gaze
[20]	Estimation of eye gaze without an eye-tracking device	Inertial sensors	GBR	Time-series head orientation data from the inertial sensors of the VR headset	Current eye gaze
[21]	Proof of concept for the use of head and hand motion to estimate eye gaze	Inertial sensors	Various ML models	Head and hand pose motion data from the inertial sensors of the VR headset and controllers	Current eye gaze
[23]	Noninvasive eye movement monitoring with eyes open or closed	Radiooculography (ROG) sensing units	Vision Transformer model	ROG signals from near-eye sensing units	Current eye gaze
[24]	3-D point-of-gaze estimation	Infrared lights and binocular cameras	Regression model	Images of infrared-light-illuminated eyes captured by infrared cameras	Current eye gaze
[25]–[27]	Appearance-based gaze estimation	Distributed cameras	ML models	Images of monitored person(s) from distributed cameras	Current eye gaze
This work	Prediction of future eye gaze	Inertial sensors	Various ML models in one and two stages	Head motion data from the inertial sensors of the VR headset	Future eye gaze

left eye region. The resulting image is then processed using image preprocessing methods and a trained convolutional neural network (CNN) for user identification as well as blink and gaze detection. Yang et al. [14] proposed using a combination of the image from the VR screen and a captured image of the reflection from the eye to obtain a rough gaze estimate. These rough estimates are then calibrated to obtain a more accurate estimate using the head motion data supplied by the headset's inertial sensors. Greenwald et al. [15] also proposed using the front-facing smartphone camera in a VR headset to capture images of the corneal reflection of on-screen content. As the eye moves, the location and features change. These changes are used to estimate the user's gaze position. However, these approaches are applicable only for the limited set of VR headsets that attach to a smartphone. Another drawback is that they capture only one eye. Some users may have asymmetric coordination of the left and right eye gazes, making these approaches less able to predict the eye gaze of both eyes. Research has also been conducted on three-dimensional eye gaze estimation, which



(a) User 1, T = 0 ms (b) User 1, T = 100 ms (c) User 1, T = 200 ms (d) User 1, T = 300 ms

(e) User 2, T = 0 ms (f) User 2, T = 100 ms (g) User 2, T = 200 ms (h) User 2, T = 300 ms

(i) User 3, T = 0 ms (j) User 3, T = 100 ms (k) User 3, T = 200 ms (l) User 3, T = 300 ms

(m) User 4, T = 0 ms (n) User 4, T = 100 ms (o) User 4, T = 200 ms (p) User 4, T = 300 ms

(q) User 5, T = 0 ms (r) User 5, T = 100 ms (s) User 5, T = 200 ms (t) User 5, T = 300 ms

FIGURE 1. Scatter plots of head orientation vs. eye gaze for different users and various anticipation times.

includes estimating gaze depth [23]. A recent advancement in this field is appearance-based gaze estimation, which employs distributed cameras in less-constrained or unconstrained environments for various monitoring applications, such as monitoring drivers [24], retail customers [25], and patients [26], among others.

To reduce the hardware cost and form factor of video-oculography-based eye tracking, some research works have focused on photosensor oculography, in which photodiodes are used instead of video cameras [27], [28], [29]. In this method, an infrared light source illuminates the eyes, and the nature of the reflection difference between the cornea and the sclera is utilized to estimate the eyes' position. The infrared sources and sensors can be integrated into spherical glasses of a reasonable size. Thus, some researchers [30], [31] have suggested that this method is suitable for embedding into VR headsets. Li et al. [6] proposed using infrared LEDs and photodiodes as the illumination sources and reflection detectors, respectively, and utilized a CNN to obtain the eye gaze position. They additionally proposed a design variant in which infrared LEDs are used as both the illumination sources and reflection sensors, thus eliminating the need for photodiodes. Li et al. [9], [10], [11] also proposed using the existing light from the VR screen as the source of illumination. Sixteen photodiodes were installed around the VR lens to measure the screen light reflected in different directions. However, this solution has the drawback that ambient light (fluorescent light, direct and indirect sunlight) can affect its performance. Massin et al. [8] proposed a device using infrared LEDs and photodetectors placed on contact lenses worn by the user. The photoreceptors are illuminated by infrared LEDs placed in front of the eye on a glasses frame. This solution carries the burden of requiring the user to wear contact lenses. Sun et al. [16] used acoustic ultrasound sensors instead of optical sensors and made use of time-of-flight information, still applying a basic principle similar to that of photosensor oculography. However, ultrasound waves are susceptible to interference from both ambient noise and each other.

Approaches using other biosignals and corresponding sensors to detect them have also been proposed. Electrooculography (EOG) measures the corneoretinal standing potential around the eyes. This method requires placing several electrodes near the eyes of the user and thus is more suitable for recording eye movements for medical purposes [32]. Magnetooculography relies on the scleral search coil method, in which a small coil in a specialized contact lens is placed in the eye. When the eyes are moving, the sclera and the muscles around the eyes create a magnetic field that is picked up by the coil. The magnetic field signal is then processed to determine the position of the eyes. This method is also unsuitable for everyday use because putting on the device requires professional medical help and often requires a local anesthetic [33]. Zhang and Kan [22] demonstrated that radiooculography (ROG) can serve as an alternative to EOG. ROG employs radio-frequency (RF) signals to noninvasively monitor the activity of the internal eye muscles, regardless of whether the eyes are open or closed.

Eye tracking using inertial sensors alone is the cheapest method of all and is also suitable for VR since all VR headsets have built-in inertial sensors for head orientation tracking. Sitzmann et al. [18] proposed deriving a saliency map indicating the region where the user's eyes will tend to focus first using the longitudinal head velocity alone. However, the proposed prediction is valid only for slow head speeds (below 19.6 degrees per second). Murakami and Mitsugami [19] collected eye-tracking data from a specialized device and head inertial data from VR headset sensors and then trained an ML model using a method called gradient boosting regression (GBR). Emery et al. [20] also employed an ML approach, using head and hand motion data together with the expected saliency maps of the VR scene. A mathematical model was proposed by Mitsugami et al. [17] in which the relationship between the eye and the head is modeled as a dynamic system of two balls connected by a spring. A system of differential equations can then be derived based on this model and solved using the random sampling consensus (RANSAC) algorithm.

A comparative summary of the related research works and our work is given in Table 2. In particular, previous research works that have used only inertial sensors for eye tracking have addressed only the limited problem of predicting the current eye gaze of the user. There has yet been no attempt in the literature to predict the future eye gaze without extra hardware. The distinctive feature of our work is that it considers a wide range of problems, including the ultimate problem of future eye gaze prediction, and presents corresponding solution designs and performance evaluations based on comprehensive combinations of ML models for the considered problems.

III. OBSERVATIONS ON THE PREDICTABILITY OF EYE MOVEMENTS

We observe the correlation between eye gaze and head orientation from the following two perspectives:

- Current head orientation vs. current eye gaze.
- Current head orientation vs. future eye gaze.

To this end, we draw scatter plots between the recorded head orientation at t (the horizontal axis) and the recorded eye gaze at $t + T$ (the vertical axis) for each user in Fig. 1, where T is the anticipation time for future eye gaze prediction. We consider $T = 0$, 100, 200, and 300 ms, where $T = 0$ ms corresponds to the case of current head orientation vs. current eye gaze, while $T > 0$ corresponds to the case of current head orientation vs. future eye gaze. The experimental setup for data collection was the same as that described in Section V-A.

First, the plots for $T = 0$ (the first subfigure in each row) suggest that the two variables are linearly correlated. This observation is made for all five users. That is, a strong correlation between the variables is observed, thus implying that eye gaze can be predicted from head orientation. Previous studies also support this preliminary conclusion [34], [35].

For $T > 0$, the two variables still show a relationship, but it is no longer linear, instead forming a *lemniscate* shape. This shape becomes more noticeable as T increases. A similar pattern is observed for all users, but their shapes at a specific T are all different from each other. This lemniscate relationship is caused by the fact that a user may rotate his/her head either left or right, so the future eye gaze can lie on either side of the current head orientation. Our observations reveal that as the latency increases, it becomes impossible to predict the eye gaze from a single head orientation sample; instead, the motion path (i.e., time-series data) of the head orientation becomes necessary for prediction. In addition, the plots show that user-specific identification of the relationship between

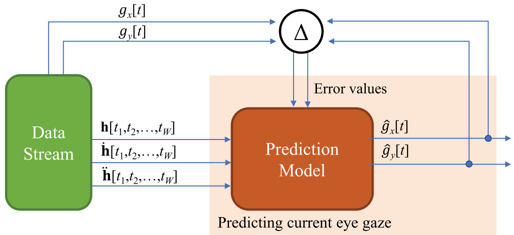
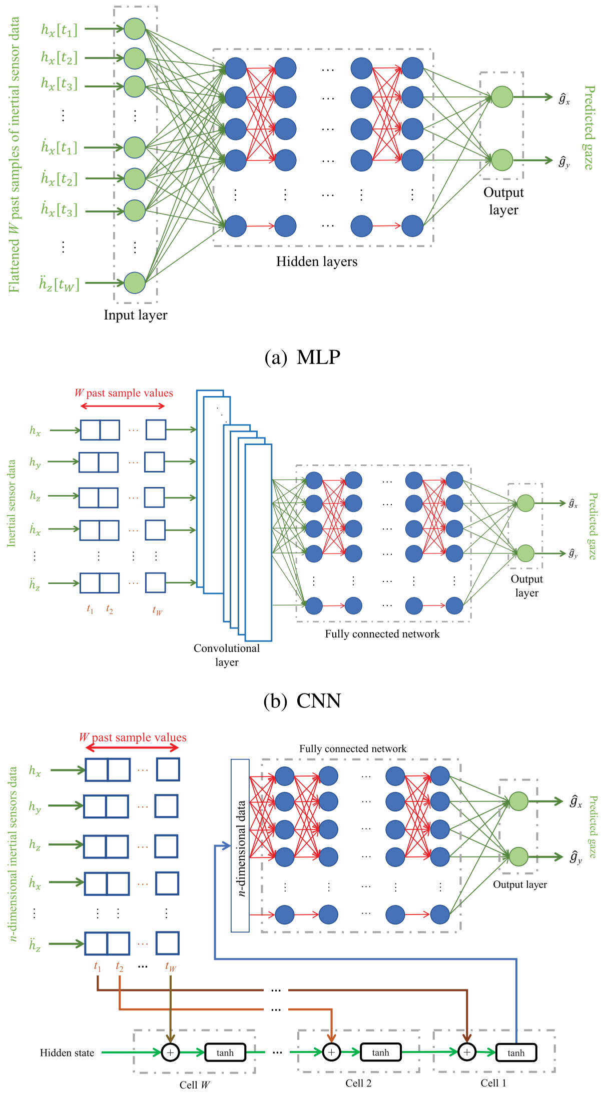

$g_x[t]$
$g_y[t]$
Δ
Error values
Data Stream
$\mathbf{h}[t_1,t_2,\ldots,t_W]$
$\dot{\mathbf{h}}[t_1,t_2,\ldots,t_W]$
$\ddot{\mathbf{h}}[t_1,t_2,\ldots,t_W]$
Prediction Model
$\hat{g}_x[t]$
$\hat{g}_y[t]$
Predicting current eye gaze

FIGURE 2. **Data flow of a prediction model for solving Problem 1 in the training phase.**



(a) MLP

(b) CNN

(c) RNN

FIGURE 3. **MLP, CNN, and RNN models for Problem 1.**

the two variables is needed due to the user-dependent nature of the correlation pattern.

IV. PROBLEM DEFINITION AND SOLUTIONS

We formulate each problem in detail and develop prediction models for each. Regarding notation, we denote the value of the variable x at time t by $x[t]$. To denote a time series of x at times $t_1, t_2, \cdots$, we use $x[t_1, t_2, \cdots]$.

A. DEFINITION OF THE GENERAL PREDICTION PROBLEM

We assume that the head orientation captured at time t by the inertial sensor unit is given by $\mathbf{h}[t] = \langle h_x[t], h_y[t], h_z[t]\rangle$, where h_x, h_y and h_z are the azimuthal (yaw), polar (pitch), and banking (roll) angles, respectively, in an Euler-angle rotational coordinate system. The eye gaze is defined as the direction along which the user is looking in the field of view (FOV), which is the sum of the head orientation and the eye direction angle. We denote the eye gaze at t by $\mathbf{g}[t] = \langle g_x[t], g_y[t]\rangle$. The horizontal eye gaze g_x is the sum of the head's yaw orientation and the eye's horizontal direction angle. Similarly, the vertical eye gaze g_y is the sum of the pitch orientation of the head and the vertical angle of the eye direction.

We wish to predict the user's eye gaze at $t+T$ ($T \geq 0$) from the information available at t. Accordingly, the predicted eye gaze, denoted by $\hat{\mathbf{g}}[t+T] = \langle \hat{g}_x[t+T], \hat{g}_y[t+T]\rangle$, can be defined as a function of a window of sensor data samples for head orientation, angular velocity, acceleration, and prior eye gaze. Thus, we have

$$\begin{aligned}\hat{\mathbf{g}}[t+T] = f_{\theta,T}(&\mathbf{h}[t_1, t_2, \cdots, t_W];\\ &\dot{\mathbf{h}}[t_1, t_2, \cdots, t_W];\\ &\ddot{\mathbf{h}}[t_1, t_2, \cdots, t_W];\\ &\mathbf{g}[t_1, t_2, \cdots, t_W])\end{aligned} \tag{1}$$

and

$$t_k = t - (k-1)\tau, \quad k = 1, 2, \cdots, W, \tag{2}$$

where θ represents the model parameters of the function f; $\dot{\mathbf{h}}$ and $\ddot{\mathbf{h}}$ are the angular velocity and acceleration, respectively, of the head orientation as captured by the gyroscope and accelerometer, respectively; τ is the time interval between consecutive data samples; and W is the window length of the input data samples. Then, the prediction error is defined as

$$\mathbf{e}[t+T] = \hat{\mathbf{g}}[t+T] - \mathbf{g}[t+T]. \tag{3}$$

For the prediction of N samples, we calculate the mean absolute error (MAE) of prediction as

$$\bar{\mathbf{e}} = \frac{1}{N}\sum_{k=1}^{N} |\mathbf{e}[k\tau]|. \tag{4}$$

The function (model) f and its parameter set θ need to be found so as to minimize the MAE $\bar{\mathbf{e}}$.

B. SOLUTIONS FOR PROBLEM 1: PREDICT THE CURRENT EYE GAZE FROM PAST HEAD MOTION DATA

The objective of Problem 1 is to find the model f_θ for $T = 0$, without input data for $\mathbf{g}$, such that

$$\begin{aligned}\hat{\mathbf{g}}[t] = f_{\theta,T=0}(&\mathbf{h}[t_1, t_2, \cdots, t_W];\\ &\dot{\mathbf{h}}[t_1, t_2, \cdots, t_W];\\ &\ddot{\mathbf{h}}[t_1, t_2, \cdots, t_W]).\end{aligned} \tag{5}$$

A model architecture for solving Problem 1 is illustrated in Fig. 2.

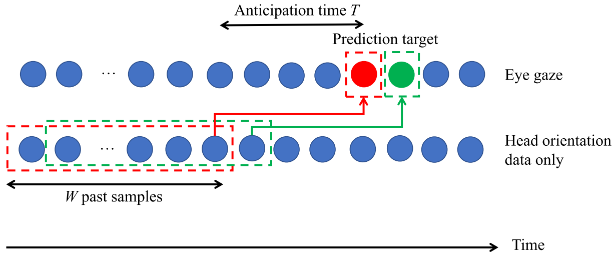


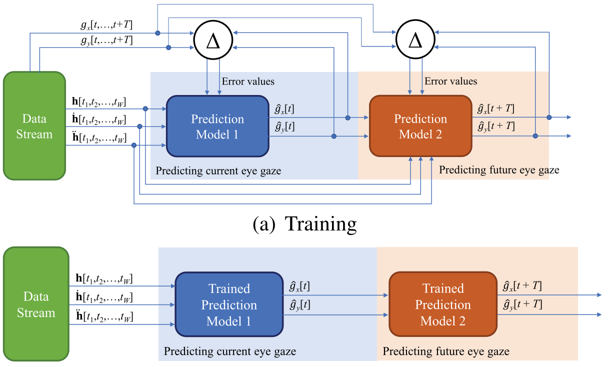
FIGURE 4. Relationship between the past data samples and predictions over time for Problem 3.

The simplest approach to solve this problem is to assume that the eye gaze is always in the center of the FOV, i.e., $\hat{\mathbf{g}}[t] = \langle h_x[t], h_y[t]\rangle$; this is the approach adopted for foveated rendering without eye tracking in current headset devices [36]. We call this approach *Center* in the evaluation section. Another simple approach is to assume that f is a linear function. This is based on studies [34], [35], whose results indicate a linear relationship when a human is stationary. Okada et al. [37] also reported that this linear relationship tends to hold for humans who are walking. With this linear approach, which we call *linear fit (LF)* regression, we obtain $\hat{\mathbf{g}}[t] = \alpha\langle h_x[t], h_y[t]\rangle$, where the coefficient α is found using least-squares linear regression. Another approach is to model the head–eye relationship as an nth-order dynamic system:

$$\hat{\mathbf{g}}[t] = \alpha_3\langle \ddot{h}_x[t], \ddot{h}_y[t]\rangle + \alpha_2\langle \dot{h}_x[t], \dot{h}_y[t]\rangle + \alpha_1\langle h_x[t], h_y[t]\rangle + \alpha_0, \tag{6}$$

where the coefficients are sought using RANSAC [17].

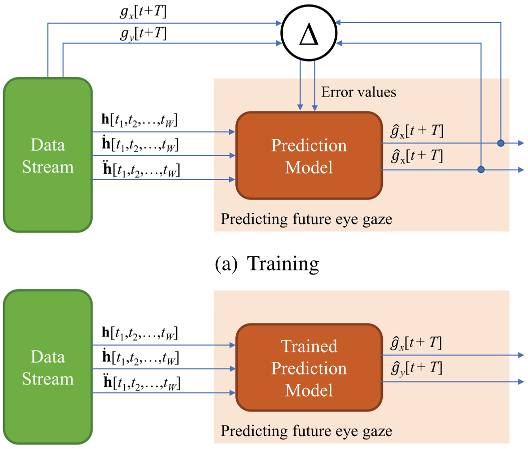
For ML-based approaches, we consider multilayer perceptron (MLP), GBR, CNN, recurrent neural network (RNN) and long short-term memory (LSTM) models. Fig. 3 illustrates the MLP, CNN, and RNN models. For the MLP model, we flatten all of the time-series input data $\mathbf{h}$, $\dot{\mathbf{h}}$, $\ddot{\mathbf{h}}$, and $\mathbf{g}$ into a single array. The input data, once flattened, are passed through multiple hidden layers within the MLP model, ultimately yielding two distinct output values, one corresponding to $\hat{g}_x$ and the other to $\hat{g}_y$. GBR-based prediction was proposed in [19] for finding the model parameters of the system dynamics model in Eq. (6). For the CNN, the time series from the various inertial sensors are separately input into the model without flattening. These input data series are filtered in the convolutional layers to extract the features of the data. After the data features are obtained from the convolutional layers, they are input into a subsequent fully connected network. In the RNN model, W cells are utilized, with each cell receiving input data from a specific time point and the preceding cell. The output from this chain of cells, which has the same dimensions as the initial input, is then fed into a fully connected network that produces $\hat{g}_x$ and $\hat{g}_y$. The structure of the LSTM model also closely resembles this configuration. For all these models, the loss function is the MAE $\bar{\mathbf{e}}$, as given in Eq. (4), where N is the number of training samples.


$g_x[t,\ldots,t{+}T]$
$g_y[t,\ldots,t{+}T]$
Δ
Δ
Error values
Error values
Data Stream
$\mathbf{h}[t_1,t_2,\ldots,t_W]$
$\dot{\mathbf{h}}[t_1,t_2,\ldots,t_W]$
$\ddot{\mathbf{h}}[t_1,t_2,\ldots,t_W]$
Prediction Model 1
$\hat{g}_x[t]$
$\hat{g}_y[t]$
Prediction Model 2
$\hat{g}_x[t+T]$
$\hat{g}_y[t+T]$
Predicting current eye gaze
Predicting future eye gaze
Data Stream
$\mathbf{h}[t_1,t_2,\ldots,t_W]$
$\dot{\mathbf{h}}[t_1,t_2,\ldots,t_W]$
$\ddot{\mathbf{h}}[t_1,t_2,\ldots,t_W]$
Trained Prediction Model 1
$\hat{g}_x[t]$
$\hat{g}_y[t]$
Trained Prediction Model 2
$\hat{g}_x[t+T]$
$\hat{g}_y[t+T]$
Predicting current eye gaze
Predicting future eye gaze

(a) Training

(b) Inference

FIGURE 5. Training and inference phases of the two-stage approach.


$g_x[t{+}T]$
$g_y[t{+}T]$
Δ
Error values
Data Stream
$\mathbf{h}[t_1,t_2,\ldots,t_W]$
$\dot{\mathbf{h}}[t_1,t_2,\ldots,t_W]$
$\ddot{\mathbf{h}}[t_1,t_2,\ldots,t_W]$
Prediction Model
$\hat{g}_x[t+T]$
$\hat{g}_x[t+T]$
Predicting future eye gaze
Data Stream
$\mathbf{h}[t_1,t_2,\ldots,t_W]$
$\dot{\mathbf{h}}[t_1,t_2,\ldots,t_W]$
$\ddot{\mathbf{h}}[t_1,t_2,\ldots,t_W]$
Trained Prediction Model
$\hat{g}_x[t+T]$
$\hat{g}_y[t+T]$
Predicting future eye gaze

(a) Training

(b) Inference

FIGURE 6. Training and inference phases of the single-stage approach.

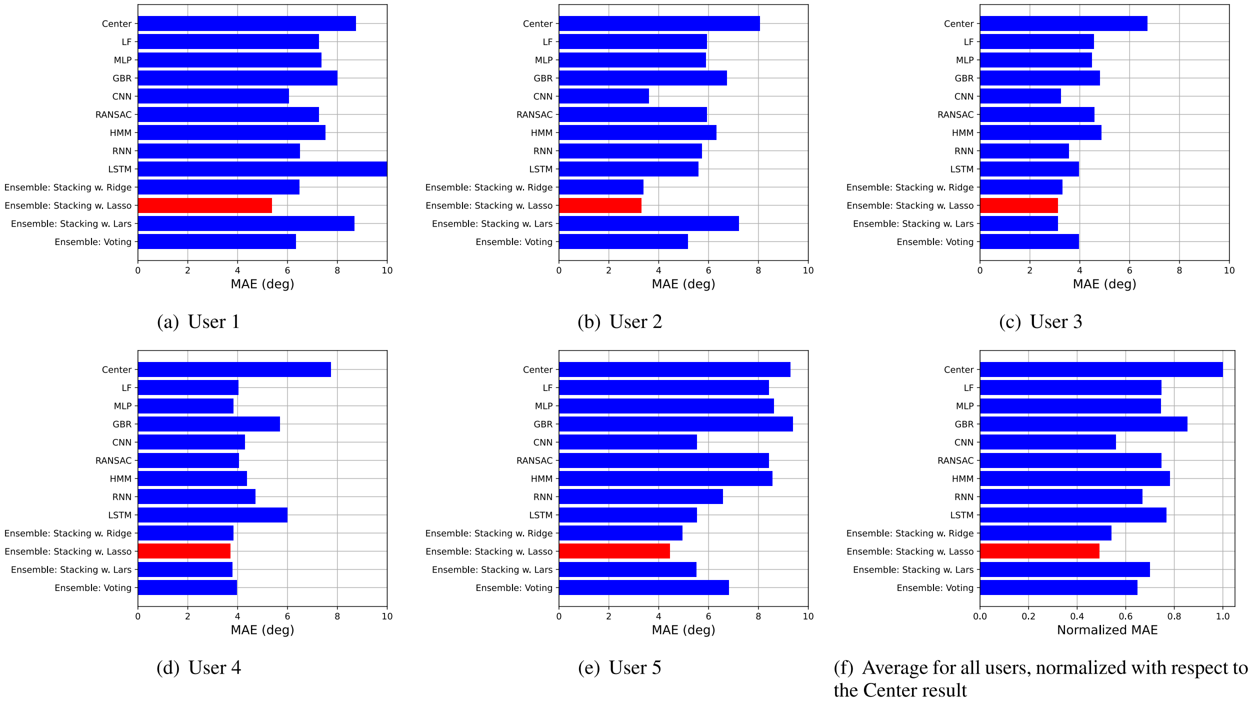
C. SOLUTIONS FOR PROBLEM 2: PREDICT THE FUTURE EYE GAZE FROM PAST EYE GAZE AND HEAD MOTION DATA

The objective of Problem 2 is to find the model f_θ for $T > 0$, with input data for $\mathbf{g}$, such that

$$\begin{aligned}\hat{\mathbf{g}}[t+T] = f_{\theta,T}(&\mathbf{h}[t_1, t_2, \cdots, t_W];\\ &\dot{\mathbf{h}}[t_1, t_2, \cdots, t_W];\\ &\ddot{\mathbf{h}}[t_1, t_2, \cdots, t_W];\\ &\mathbf{g}[t_1, t_2, \cdots, t_W]).\end{aligned} \tag{7}$$

We first consider three approaches that do not make use of ML:

- *No prediction (NOP)*: In this approach, the future eye gaze is simply assumed to be the same as the current eye gaze, i.e., $\hat{\mathbf{g}}[t+T] = \mathbf{g}[t]$, which is true when the user's eye gaze is stationary. We consider this approach as a baseline to evaluate the gains of other solutions.
- *Constant rate prediction (CRP)*: This approach assumes that the angular velocity of the user's head ($\dot{\mathbf{h}}$) and the relative eye gaze both remain unchanged for the anticipation time T. Accordingly, the head shift during



(a) User 1

(b) User 2

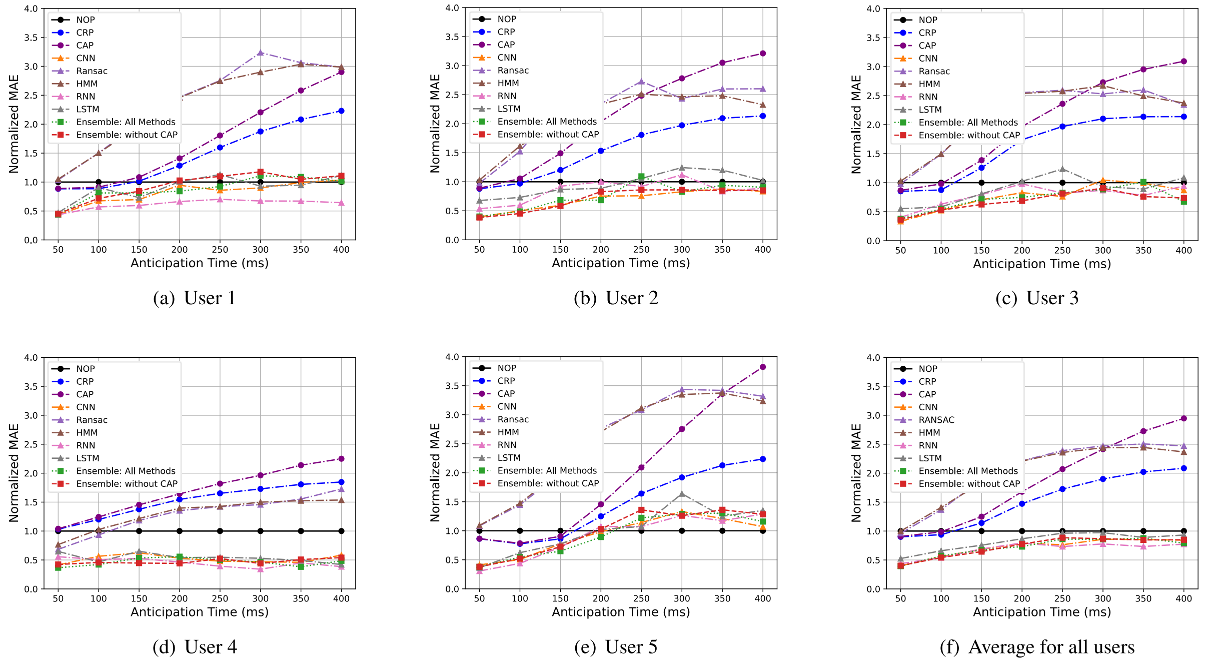
(c) User 3

(d) User 4

(e) User 5

(f) Average for all users, normalized with respect to the Center result

FIGURE 7. MAE performance of the prediction models for Problem 1.



(a) User 1

(b) User 2

(c) User 3

(d) User 4

(e) User 5

(f) Average for all users

FIGURE 8. Normalized MAE performance of the prediction models for Problem 2.

T is obtained as $\dot{\mathbf{h}}[t]T$, and we have $\hat{\mathbf{g}}[t+T] = \mathbf{g}[t] + \langle \dot{h}_x[t], \dot{h}_y[t] \rangle T$.

- *Constant acceleration prediction (CAP)*: This approach assumes that the angular acceleration of the user's head ($\ddot{\mathbf{h}}$) and the relative eye gaze both remain unchanged for the anticipation time T. Accordingly, the head shift during T is obtained as $\dot{\mathbf{h}}[t]T + \frac{1}{2}\ddot{\mathbf{h}}[t]T^2$, and we have $\hat{\mathbf{g}}[t+T] = \mathbf{g}[t] + \langle \dot{h}_x[t], \dot{h}_y[t] \rangle T + \frac{1}{2}\langle \ddot{h}_x[t], \ddot{h}_y[t] \rangle T^2$.

For ML-based approaches, an architecture similar to that for Problem 1 is considered, with the addition of the current eye gaze as an input to the model. Moreover, the output is not for the current time but rather for a future time advanced by the anticipation time T, i.e., the future eye gaze at time $t + T$ is predicted at time t. The remaining components of the model structures remain unchanged from those for Problem 1. We also employ ensemble methods by combining

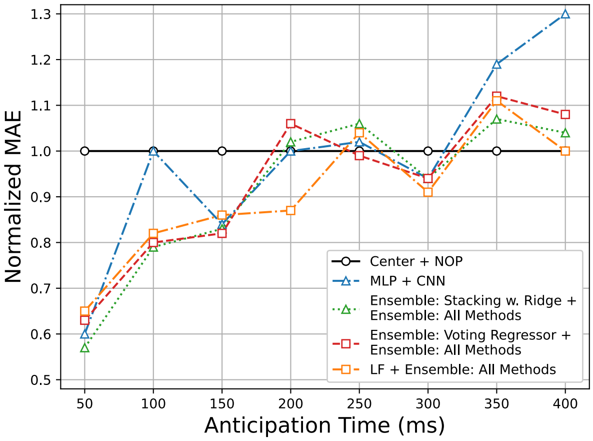


FIGURE 9. Normalized MAE performance of the two-stage models for Problem 3 with varying anticipation times.

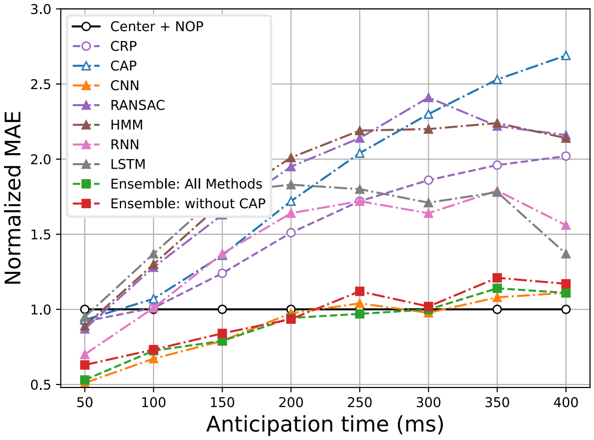


FIGURE 10. Normalized MAE performance of the single-stage models for Problem 3 with varying anticipation times.

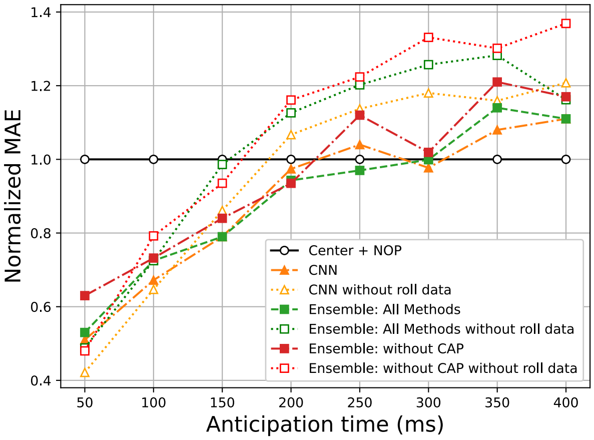


FIGURE 11. Normalized MAE performance of the top three models with and without roll input data for Problem 3 with varying anticipation times.

all of the methods described above to determine the best model output to use.

D. SOLUTIONS FOR PROBLEM 3: PREDICT THE FUTURE EYE GAZE FROM PAST HEAD MOTION DATA

We define Problem 3 as the problem of predicting the future gaze from past inertial sensor data. That is, the objective of Problem 3 is to find the model f_θ for $T > 0$, without input data for $\mathbf{g}$, such that

$$\begin{aligned}\hat{\mathbf{g}}[t+T] = f_{\theta,T}(&\mathbf{h}[t_1, t_2, \cdots, t_W];\\ &\dot{\mathbf{h}}[t_1, t_2, \cdots, t_W];\\ &\ddot{\mathbf{h}}[t_1, t_2, \cdots, t_W]).\end{aligned} \tag{8}$$

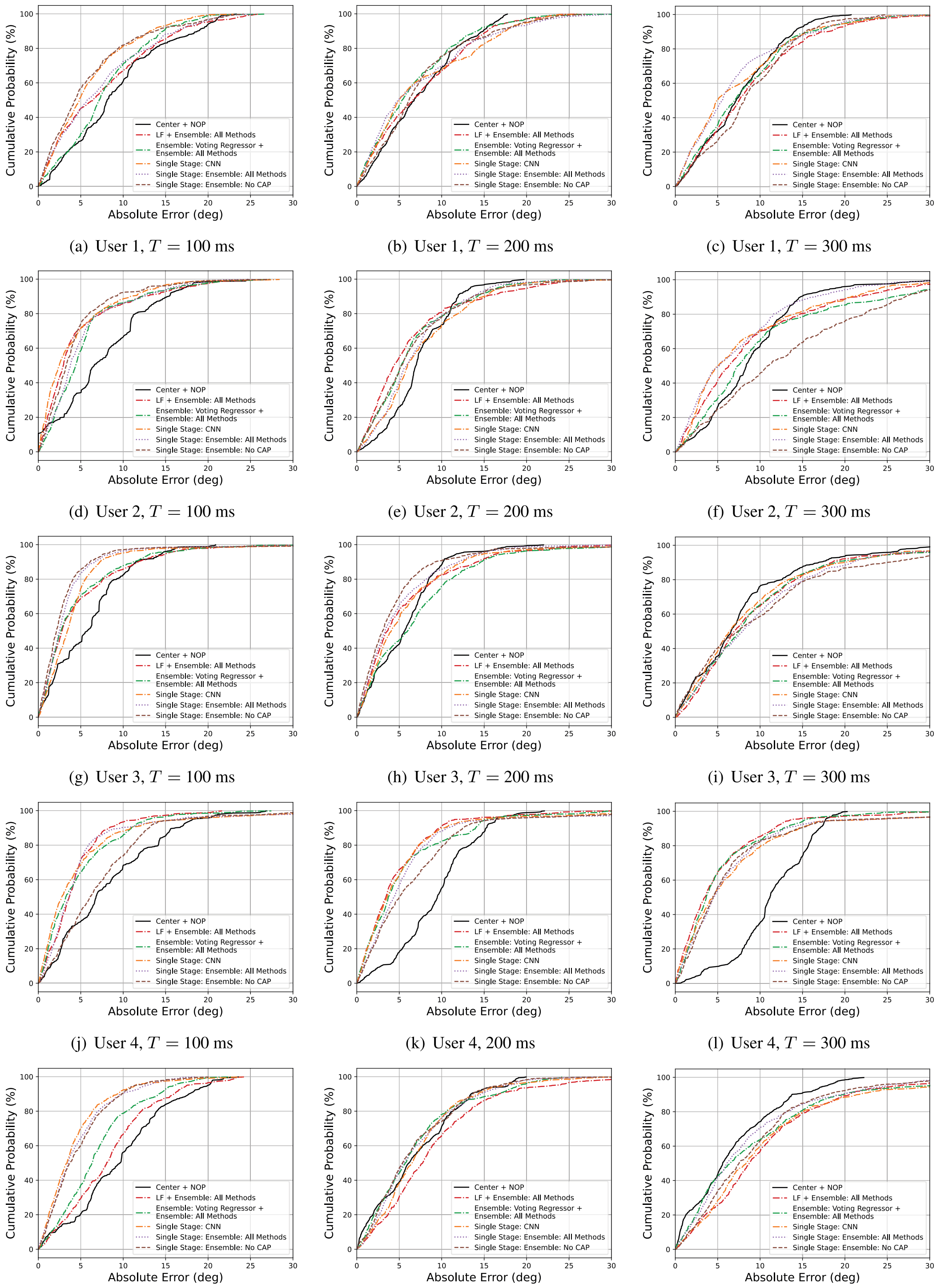
Fig. 4 illustrates the relationship between the past data samples and the predictions over time.

The first approach illustrated in Fig. 5 is to serially combine the solutions for Problems 1 and 2, which we call the *two-stage* approach. At time t and before, the first submodel (a solution for Problem 1), which we denote by $f^1_{\theta_1,T=0}$, yields predicted eye gaze samples for t and before, i.e., $\hat{\mathbf{g}}[t_1 = t], \hat{\mathbf{g}}[t_2], \cdots$. Then, to predict the final output $\hat{\mathbf{g}}[t+T]$, the second submodel (a solution for Problem 2), which is denoted by $f^2_{\theta_1,T}$, uses the predicted eye gaze samples instead of actual eye gaze samples. θ_1 and θ_2 are the parameter sets of the first and second submodels, respectively. Thus, we rewrite Eq. (8) for the two-stage approach as

$$\begin{aligned}\hat{\mathbf{g}}[t+T] &= f^2_{\theta_2,T} \circ f^1_{\theta_1,T=0}(\mathbf{h}[t_1, t_2, \cdots, t_W];\\ &\qquad\qquad \dot{\mathbf{h}}[t_1, t_2, \cdots, t_W];\\ &\qquad\qquad \ddot{\mathbf{h}}[t_1, t_2, \cdots, t_W])\\ &= f^2_{\theta_2,T}(\mathbf{h}[t_1, t_2, \cdots, t_W];\\ &\qquad \dot{\mathbf{h}}[t_1, t_2, \cdots, t_W];\\ &\qquad \ddot{\mathbf{h}}[t_1, t_2, \cdots, t_W];\\ &\qquad \hat{\mathbf{g}}[t_1, t_2, \cdots, t_W]),\end{aligned} \tag{9}$$

where $\hat{\mathbf{g}}$ is obtained from the first submodel $f^1_{\theta_1,T=0}$. In the training procedure for the two-stage approach, the initial step involves training the first submodel. Following this, the second submodel is trained, utilizing the inertial data and the output data produced by the first submodel.

The second approach, which we call the *single-stage* approach, uses only a single ML model, which is trained to directly find θ in Eq. (8). The data flows of the single-stage approach in the training and inference phases are illustrated in Fig. 6. The input data for the single-stage approach are the same as those for the first submodel in the two-stage approach. In contrast to the two-stage approach, which requires eye gaze data as input for the second model, the single-stage approach requires only head orientation data as input, based on which the single-stage approach directly outputs the predicted eye gaze. The training and inference processes for the single-stage approach are more computationally efficient than those for the two-stage approach. This is because a single-stage model requires only one step during training and propagation of the input data through only a single model during inference, whereas for a two-stage model, both submodels must be involved in both processes.



(a) User 1, $T = 100$ ms

(b) User 1, $T = 200$ ms

(c) User 1, $T = 300$ ms

(d) User 2, $T = 100$ ms

(e) User 2, $T = 200$ ms

(f) User 2, $T = 300$ ms

(g) User 3, $T = 100$ ms

(h) User 3, $T = 200$ ms

(i) User 3, $T = 300$ ms

(j) User 4, $T = 100$ ms

(k) User 4, 200 ms

(l) User 4, $T = 300$ ms

(m) User 5, $T = 100$ ms

(n) User 5, 200 ms

(o) User 5, $T = 300$ ms

FIGURE 12. CDFs of absolute prediction error samples for the top five prediction methods for Problem 3.

V. PERFORMANCE EVALUATION, ANALYSIS, AND DISCUSSION

A. EXPERIMENTAL SETUP

We used an HTC VIVE headset [38] and an aGlass (DK II) eye-tracking device [39] installed in the headset. While users played a VR program in which they looked at paintings on the walls of an art gallery (a modified version of a sample program provided with the aGlass device), both head motion and eye gaze data were recorded at 60 Hz into trace files so that all algorithms could be run with the same input data to ensure fair comparisons. We used $W = 20$ samples, corresponding to a period of 1/3 s. Throughout the experiments, the parameters of the models were configured as follows. For the LF model, we used regular least-squares linear regression

TABLE 3. Prediction performance ranking.

Method \ Anticipation time (ms)	Problem 1	Problem 2					Problem 3				
	N/A	50	100	200	300	400	50	100	200	300	400
NOP / Center / Center + NOP	13	10	8	6	6	6	17	12	6	7	**2**
LF	8	-	-	-	-	-	-	-	-	-	-
MLP	7	-	-	-	-	-	-	-	-	-	-
GBR	12	-	-	-	-	-	-	-	-	-	-
CNN	**3**	**2**	**3**	**3**	**3**	**3**	4	**2**	4	5	6
CNN w/o roll data	-	-	-	-	-	-	**1**	**1**	9	9	9
RANSAC	9	8	9	10	10	9	12	15	13	17	16
HMM	11	9	10	9	9	8	13	16	17	15	15
RNN	5	4	**1**	4	**1**	**1**	11	11	14	12	13
LSTM	10	5	5	5	5	5	16	17	16	13	11
Ensemble: w. Ridge	**2**	-	-	-	-	-	-	-	-	-	-
Ensemble: w. Lasso (All Methods)	**1**	**1**	4	**1**	**2**	**2**	5	**3**	**3**	6	5
Ensemble: w. Lasso, w/o roll data	**1**	-	-	-	-	-	**3**	4	10	10	7
Ensemble: w. Lasso, w/o CAP	**1**	**2**	**2**	**2**	4	4	8	5	**2**	8	8
Ensemble: w. Lasso, w/o CAP, w/o roll data	**1**	-	-	-	-	-	**2**	7	11	11	12
Ensemble: w. Lars	6	-	-	-	-	-	-	-	-	-	-
Ensemble: Voting	4	-	-	-	-	-	-	-	-	-	-
CRP	-	6	6	7	7	7	14	13	12	14	14
CAP	-	7	7	8	8	10	15	14	15	16	17
MLP + CNN	-	-	-	-	-	-	7	10	5	**2**	10
Ensemble: w. Ridge + Ensemble: w. Lasso	-	-	-	-	-	-	6	6	7	4	**3**
Ensemble: Voting + Ensemble: w. Lasso	-	-	-	-	-	-	9	8	8	**3**	**3**
LF + Ensemble: w. Lasso	-	-	-	-	-	-	10	9	**1**	**1**	**1**

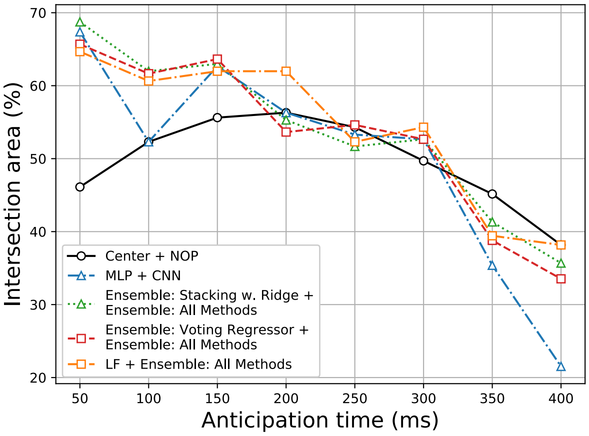


FIGURE 13. Intersection performance of foveated rendering under the two-stage models for Problem 3 with varying anticipation times.

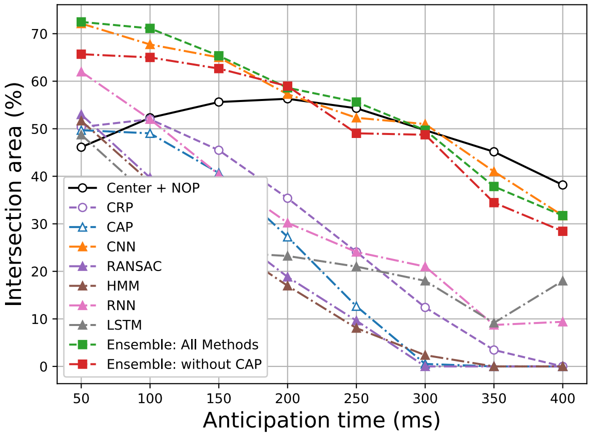


FIGURE 14. Intersection performance of foveated rendering under the single-stage models for Problem 3 with varying anticipation times.

with intercept calculation and no normalization. For the GBR model, we used one thousand estimators and a learning rate of 0.01. For the MLP model, we used five hidden layers of 100, 80, 40, 30, and 20 neurons in sequence. For the CNN model, we used 27×1 convolutional layers and a fully connected network with two hidden layers consisting of 18 and 9 neurons in the first and second layers, respectively. For both the MLP and CNN, the activation function was ReLU, the optimizer was the Adam optimizer, and the learning rate was set to 0.01. In the RANSAC model [19], a future sample is calculated as the product of the current sample and a coefficient that is derived from the RANSAC estimator. To construct a hidden Markov model (HMM) [40], the range of variation between consecutive samples was divided into 40 intervals, each representing a state within the HMM. When a change is forecast by the HMM, the future sample is determined by adding the predicted change to the current sample.

B. EVALUATION RESULTS FOR PROBLEM 1

In addition to the models previously discussed in Section IV (Center, CNN, LF, MLP, and GBR), we also consider the stacking and voting ensemble methods [41], [42]. A stacked ensemble model [43] takes the outputs of multiple models as the inputs to a meta-regressor, which then gives the final prediction result. The meta-regressors considered here are ridge regression [44], the least absolute shrinkage and selection operator (LASSO) [45], and least angle regression and shrinkage (LARS) [46], [47].

Fig. 7 shows the MAE comparison results for each user in panels (a)–(e) and the average result for all users normalized with respect to the MAE of Center in panel (f). In each panel of this figure, the bar corresponding to the best-performing method is colored red. We observe that the stacked ensemble model with the LASSO regressor consistently shows the smallest MAE for all users. For User 4, the other models achieve MAEs similar to those of the stacked ensemble and CNN models, but for the other users, the performance gap between the models is not negligible. Fig. 7(f) shows that the stacked ensemble model with the LASSO regressor achieves an MAE reduction of 50% compared to the Center model. We also see that the stacked ensemble models with different regressors all achieve MAE performance similar to that of the CNN model. This is because each stacked ensemble model uses the output of the CNN model as one of the inputs to its meta-regressor. However, the stacking ensemble approach can achieve a slightly lower MAE than the CNN model due to its collective utilization of the other models as well.

For each method, we also show the cumulative distribution functions (CDFs) of the error samples for each user and the single CDF curve for all user samples in Appendix A.

C. EVALUATION RESULTS FOR PROBLEM 2

Similar to Problem 1, the stacking ensemble approach with the LASSO regressor is also considered for Problem 2. We compare two stacked ensemble models: one that includes all of the base models and one that does not include the worst model, namely, CAP. The MAE results are compared for each user in Fig. 8(a)–(e), and Fig. 8(f) shows the average result for all users normalized with respect to NOP. The CNN, RNN, and ensemble models show similar gains for all users. For an anticipation time of 50 ms, the reduction gains of the CNN, RNN and ensemble models over NOP are as high as 50%. As the anticipation time increases, however, the gain decreases and becomes as small as approximately 10% on average for an anticipation time of 200 ms. This is because older data samples are less correlated with the future status and thus fail to provide a model with sufficient information for prediction. The LSTM model attains marginally lower gains compared to the CNN, RNN and ensemble models. On average, there is no meaningful performance difference between the two ensemble models. The CRP and CAP models show small gains only for short anticipation times and become worse than NOP for longer anticipation times, implying that their assumptions of constant velocity and acceleration are not valid, especially for long anticipation times.

The CDFs of the error samples for each user and for different anticipation times are also given in Appendix B.

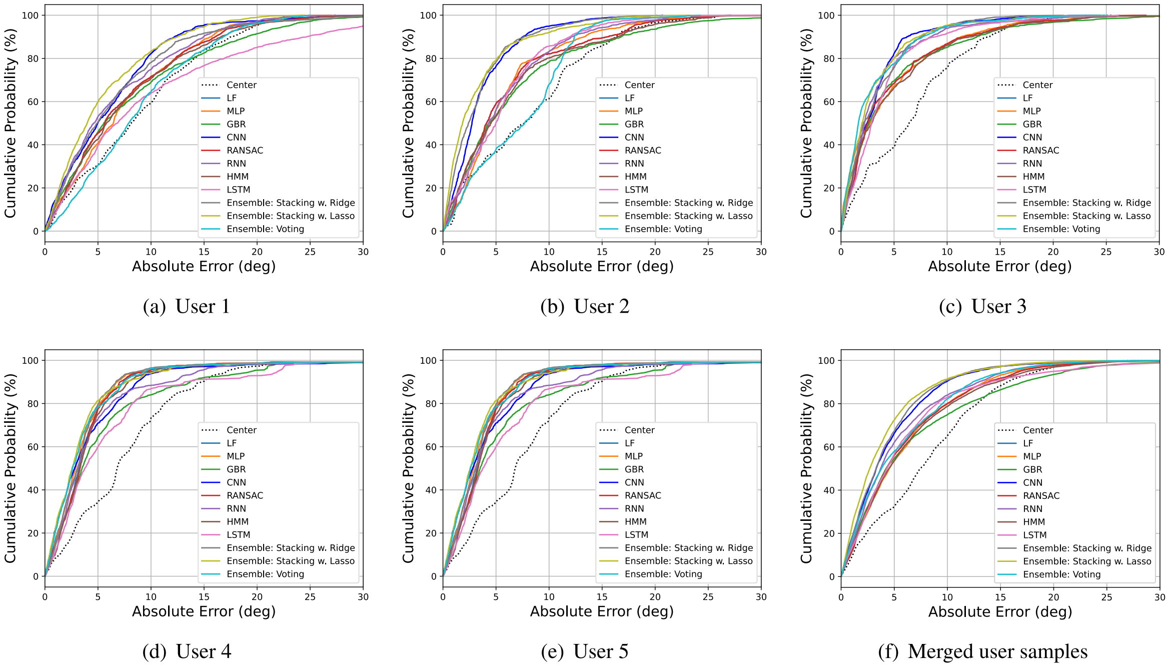
D. EVALUATION RESULTS FOR PROBLEM 3

Fig. 9 shows the comparison of the MAE results for the two-stage models normalized with respect to the MAE of Center+NOP. We construct the name of each two-stage model as the name of the first-stage model followed by the name of the second-stage model after a plus sign. We combined all of the prediction models for Problem 1 and Problem 2 into corresponding two-stage models and then sorted them based on their MAE performance. Finally, only the top five models among all combinations are shown in this figure. They achieve an MAE reduction of approximately 40% compared to Center+NOP for an anticipation time of 50 ms. This reduction decreases to approximately 20% for an anticipation time of 150 ms. For an anticipation time of 200 ms, all methods except LF+Ensemble perform similarly to Center+NOP. For an anticipation time of 250 ms, even LF+Ensemble is similar to Center+NOP. For anticipation times of 350 and 450 ms, all methods become worse than Center+NOP because predicting the future gaze becomes harder over longer anticipation times, as observed from the results for Problem 2.

The normalized MAE results for the single-stage models are compared in Fig. 10. This figure shows that the single-stage models achieve lower MAEs than the two-stage models. The CNN model and the ensemble model constructed from all base models both achieve an MAE reduction of approximately 50% compared to Center+NOP for an anticipation time of 50 ms, while the two-stage models achieve reductions of only up to 40%. For an anticipation time of 100 ms, the single-stage models still outperform the two-stage models, achieving MAE reductions of approximately 30% while the two-stage models show reductions of approximately 20%. However, the single-stage models suffer a decrease in their reduction gain with an increasing anticipation time and become similar to or worse than Center+NOP at an anticipation time of 250 ms. The ensemble approach with all base models is always better than the ensemble approach without CAP. This indicates that although CAP alone shows poor performance for Problem 2, including it in the ensemble model for Problem 3 is beneficial for MAE reduction.

We also conducted an investigation to determine the impact of roll input data on the prediction performance. The normalized MAE results for the top three single-stage models with and without roll input data are compared in Fig. 11. The models without roll data as input slightly outperform their counterparts with roll input data at an anticipation time of 50 ms. This may be because the yaw and pitch input data have a robust correlation with the eye gaze, providing sufficient information for prediction, whereas roll data function more as noise rather than contributing valuable information. However, as the anticipation time increases beyond 50 ms, the models without roll data begin to perform worse than those with roll data, with an increasing performance gap between the two model types. This implies that roll data can provide meaningful information for prediction as the correlation of the yaw and pitch data with the eye gaze becomes weaker.

Fig. 12 shows the CDFs of the absolute prediction error samples for each user. This figure includes the five best-performing models for Problem 3 and Center+NOP as a baseline. For an anticipation time of 100 ms, the single-stage models outperform the two-stage models for Users 1, 3, and 5 while showing similar performance for



(a) User 1

(b) User 2

(c) User 3

(d) User 4

(e) User 5

(f) Merged user samples

FIGURE 15. CDFs of the absolute prediction errors for Problem 1.

Users 2 and 4. All the models achieve MAE reductions compared to Center+NOP for all error samples. For an anticipation time of 200 ms, however, the prediction methods do not achieve significant gains over Center+NOP, especially for Users 1 and 5. For Users 2, 3, and 4, even the best method outperforms Center+NOP for only some of the error samples. For an anticipation time of 300 ms, prediction is still beneficial for Users 2 and 4, but for the other users, prediction performs similarly to Center+NOP (User 1) or even worse (Users 3 and 5).

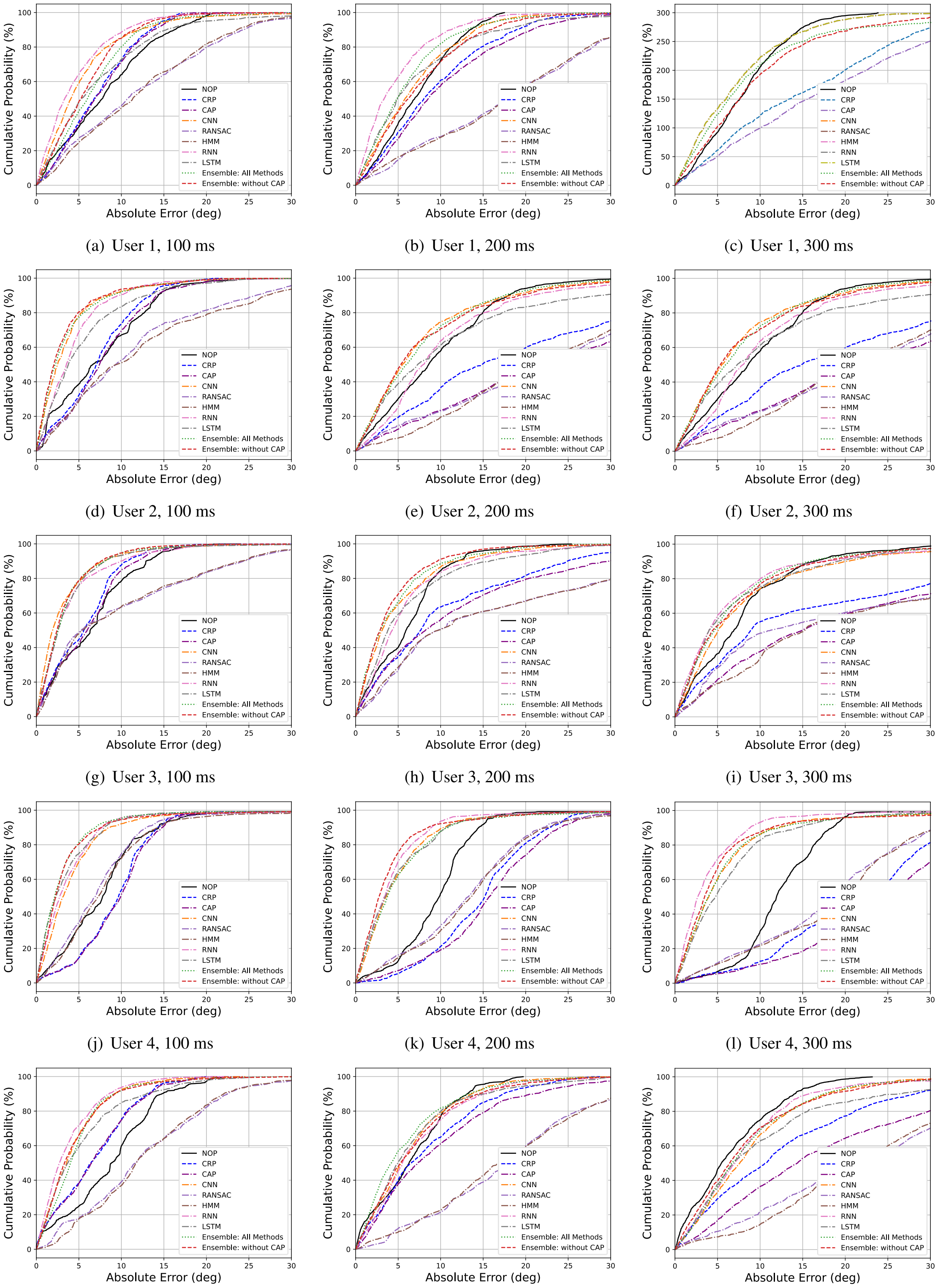
Table 3 presents the rank assigned to each method. We summarize the conclusions drawn from our experimental results as follows:

- For Problem 1, the CNN model generally delivers the best performance among all single models, while the ensemble approach, especially with LASSO regression, outperforms all other methods.
- For Problem 2, the CNN and RNN models, along with the ensemble approach, generally exhibit strong performance, with gains over NOP that decrease as the anticipation time increases. The other models underperform compared to NOP except at a short anticipation time of 50 ms.
- For Problem 3, the single-stage approach slightly outpaces the two-stage approach for a short anticipation time, but the two-stage approach becomes superior for longer anticipation times. Among the single-stage methods, the CNN model and the ensemble models generally demonstrate the best performance, with gains over NOP that again decrease as the anticipation time increases. The other models underperform compared to NOP except at a short anticipation time of 50 ms.

E. EVALUATION RESULTS FOR FOVEATED RENDERING

We conducted experiments on foveated rendering to assess the effectiveness of the proposed eye gaze prediction solutions in terms of VR service quality. The region of interest (ROI) in the user viewport, which provides the most visual information to the human visual system, should be displayed at high resolution. In foveated rendering, the region around the predicted eye gaze point, rather than the central region, is rendered at high resolution, with the expectation that the ROI will be filled with high-resolution pixels when the rendered VR image is displayed to the user. As the error of eye gaze prediction increases, the difference between the high-resolution area produced through foveated rendering and the ROI also increases, resulting in more low-resolution pixels being visible to the user in the ROI. The degree of this difference can be quantified by computing the intersection area between the high-resolution region produced through foveated rendering and the ROI, which reflects the percentage of high-resolution pixels within the ROI. In these experiments, the ROI was defined as a circular area centered around the true eye gaze, occupying 18% of the viewport. The high-resolution region produced through foveated rendering was designed as an equally sized circular area, in accordance with NVIDIA's basic default settings for foveated rendering [48]. This region was centered on the predicted eye gaze. Therefore, perfect eye gaze prediction should result in an intersection area of 100%.

The intersection areas achieved under the two-stage and single-stage models are shown in Figs. 13 and 14, respectively, for varying anticipation times. For an anticipation time of 50 ms, the two-stage models achieve intersection areas of 64% to 68%, which is approximately 20% higher than that of Center+NOP. The maximum gain



(a) User 1, 100 ms

(b) User 1, 200 ms

(c) User 1, 300 ms

(d) User 2, 100 ms

(e) User 2, 200 ms

(f) User 2, 300 ms

(g) User 3, 100 ms

(h) User 3, 200 ms

(i) User 3, 300 ms

(j) User 4, 100 ms

(k) User 4, 200 ms

(l) User 4, 300 ms

(m) User 5, 100 ms

(n) User 5, 200 ms

(o) User 5, 300 ms

FIGURE 16. CDFs of the absolute prediction errors for Problem 2.

of the single-stage models, achieved by the CNN model and the ensemble model including all methods, is even higher, reaching an intersection area of up to 72%, although some methods (RANSAC, HMM, and LSTM) show only marginal gains. As the anticipation time increases, the gains of all prediction methods decrease due to increasing prediction error. Both the two-stage and single-stage methods begin to achieve intersection areas similar to that of Center+NOP at an anticipation time of 200 ms (although LF+Ensemble still achieves a 6% gain over Center+NOP). For an anticipation time longer than 300 ms, all prediction methods achieve a worse intersection area than Center+NOP due to excessive prediction error.

VI. CONCLUSION

We developed eye-tracking solutions using only inertial sensors for the three time-series regression problems of predicting (1) the current eye gaze using past head orientation data, (2) the future eye gaze using past head orientation and eye gaze data, and (3) the future eye gaze using past head orientation data only. We solved the first and second problems using various ML models and developed two approaches to solutions for the final problem: two-stage and single-stage approaches. In the two-stage approach, two ML models are combined in series, one for the first problem and the other for the second problem. In contrast, the single-stage solutions use a single model to predict the future eye gaze directly from past head orientation data. We evaluated the proposed solutions based on real eye-tracking traces captured from a VR headset for multiple test players, considering various combinations of ML models. The results showed that prediction models are effective for anticipation times of up to a few hundred milliseconds and that the single-stage approach outperforms the two-stage approach.

APPENDIX A
CDFs OF THE ABSOLUTE PREDICTION ERRORS FOR PROBLEM 1

For each method, we show the cumulative distribution functions (CDFs) of the error samples for each user in Fig. 15(a)–(e) and the single CDF curve for all user samples in Fig. 15(f). Similar to the results presented in the main text, the CDFs also show that the stacked ensemble and CNN models outperform all other models for all samples. In Fig. 7, LF achieves a lower MAE than Center, but Fig. 15 reveals that LF has higher errors than Center for a large number of samples, i.e., approximately 15% of the samples in Fig. 15(f).

APPENDIX B
CDFs OF THE ABSOLUTE PREDICTION ERRORS FOR PROBLEM 2

For each user, Fig. 16 shows the CDFs of the error samples for anticipation times T of 100, 200, and 300 ms. For $T = 100$ ms, all models outperform NOP except for User 4. In particular, the CNN model significantly outperforms NOP for all samples and all users. The ensemble approach yields curves similar to those for the CNN model except for User 1. For $T = 200$ ms, the gap between the prediction models and NOP becomes much smaller. The CNN and ensemble models still outperform NOP, but the CNN actually results in worse error samples than NOP for User 1. CRP and CAP begin to be outperformed by NOP, consistent with our observations in the main text. For $T = 300$ ms, the performance ranking of the methods differs for different users. For User 1, the ensemble approach still outperforms NOP for over 80% of samples, but the CNN is worse than NOP for approximately 70% of samples. For Users 2, 3, and 4, the CNN and ensemble models outperform NOP for the majority of samples, but for User 5, all methods are worse than NOP, which means that prediction is actually detrimental.

REFERENCES

[1] B. T. Carter and S. G. Luke, "Best practices in eye tracking research," *Int. J. Psychophysiol.*, vol. 155, pp. 49–62, Sep. 2020.

[2] A. Patney, M. Salvi, J. Kim, A. Kaplanyan, C. Wyman, N. Benty, D. Luebke, and A. Lefohn, "Towards foveated rendering for gaze-tracked virtual reality," *ACM Trans. Graph.*, vol. 35, no. 6, pp. 1–12, Nov. 2016.

[3] V. Clay, P. König, and S. U. König, "Eye tracking in virtual reality," *J. Eye Movement Res.*, vol. 12, no. 1, p. 3, Apr. 2019.

[4] *Preparing for a Cloud AR/VR Future (White Paper)*, Huawei, Shenzhen, China, 2017.

[5] B. W. Nyamtiga, A. A. Hermawan, Y. F. Luckyarno, T. Kim, D. Jung, J. S. Kwak, and J. Yun, "Edge-computing-assisted virtual reality computation offloading: An empirical study," *IEEE Access*, vol. 10, pp. 95892–95907, 2022.

[6] R. Li, E. Whitmire, M. Stengel, B. Boudaoud, J. Kautz, D. Luebke, S. Patel, and K. Akşit, "Optical gaze tracking with spatially-sparse single-pixel detectors," in *Proc. IEEE Int. Symp. Mixed Augmented Reality (ISMAR)*, Nov. 2020, pp. 117–126.

[7] D. Katrychuk, H. K. Griffith, and O. V. Komogortsev, "Power-efficient and shift-robust eye-tracking sensor for portable VR headsets," in *Proc. 11th ACM Symp. Eye Tracking Res. Appl.*, Jun. 2019, pp. 1–8.

[8] L. Massin, V. Nourrit, C. Lahuec, F. Seguin, L. Adam, E. Daniel, and J.-L. de B. de la Tocnaye, "Development of a new scleral contact lens with encapsulated photodetectors for eye tracking," *Opt. Exp.*, vol. 28, no. 19, pp. 28635–28647, 2020.

[9] T. Li, Q. Liu, and X. Zhou, "Ultra-low power gaze tracking for virtual reality," in *Proc. 15th ACM Conf. Embedded Netw. Sensor Syst.*, Nov. 2017, pp. 1–14.

[10] T. Li, Q. Liu, and X. Zhou, "Ultra-low-power gaze tracking for virtual reality," *GetMobile, Mobile Comput. Commun.*, vol. 22, no. 3, pp. 27–31, Jan. 2019.

[11] T. Li and X. Zhou, "Battery-free eye tracker on glasses," in *Proc. 24th Annu. Int. Conf. Mobile Comput. Netw.*, Oct. 2018, pp. 67–82.

[12] K. Ahuja, R. Islam, V. Parashar, K. Dey, C. Harrison, and M. Goel, "EyeSpyVR: Interactive eye sensing using off-the-shelf, smartphone-based VR headsets," in *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 2, no. 2, Jul. 2018, pp. 1–10.

[13] P. Drakopoulos, G.-A. Koulieris, and K. Mania, "Eye tracking interaction on unmodified mobile VR headsets using the selfie camera," *ACM Trans. Appl. Perception*, vol. 18, no. 3, pp. 1–20, Jul. 2021.

[14] S. Yang, Y. He, and M. Jin, "VGaze: Implicit saliency-aware calibration for continuous gaze tracking on mobile devices," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2021, pp. 1–10.

[15] S. W. Greenwald, L. Loreti, M. Funk, R. Zilberman, and P. Maes, "Eye gaze tracking with Google cardboard using Purkinje images," in *Proc. 22nd ACM Conf. Virtual Reality Softw. Technol.*, Nov. 2016, pp. 19–22.

[16] S. Sun, J. Wang, M. Zhang, Y. Yuan, Y. Ning, D. Ma, P. Niu, Y. Gong, X. Yang, and W. Pang, "Eye-tracking monitoring based on PMUT arrays," *J. Microelectromech. Syst.*, vol. 31, no. 1, pp. 45–53, Feb. 2022.

[17] I. Mitsugami, Y. Okinaka, and Y. Yagi, "Gaze estimation based on eyeball-head dynamics," in *Proc. IEEE Winter Appl. Comput. Vis. Workshops (WACVW)*, Mar. 2017, pp. 48–52.

[18] V. Sitzmann, A. Serrano, A. Pavel, M. Agrawala, D. Gutierrez, B. Masia, and G. Wetzstein, "Saliency in VR: How do people explore virtual environments?" *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 4, pp. 1633–1642, Apr. 2018.

[19] J. Murakami and I. Mitsugami, "Gaze from head: Gaze estimation without observing eye," in *Pattern Recognition* (Lecture Notes in Computer Science), vol. 12046. Cham, Switzerland: Springer, 2020, pp. 254–267.

[20] K. J. Emery, M. Zannoli, L. Xiao, J. Warren, and S. S. Talathi, "Estimating gaze from head and hand pose and scene images for open-ended exploration in VR environments," in *Proc. IEEE Conf. Virtual Reality 3D User Interfaces Abstr. Workshops (VRW)*, Mar./Apr. 2021, pp. 554–555.

[21] A. Khaldi, E. Daniel, L. Massin, C. Kärnfelt, F. Ferranti, C. Lahuec, F. Seguin, V. Nourrit, and J.-L. de B. de la Tocnaye, "A laser emitting contact lens for eye tracking," *Sci. Rep.*, vol. 10, no. 1, pp. 1–8, Sep. 2020.

[22] Z. Zhang and E. C. Kan, "Radiooculogram (ROG) for eye movement sensing with eyes closed," in *Proc. IEEE Sensors*, Oct./Nov. 2022, pp. 1–4.

[23] J. Sun, Z. Wu, H. Wang, P. Jing, and Y. Liu, "A novel integrated eye-tracking system with stereo stimuli for 3-D gaze estimation," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–15, 2023.

[24] Y. Wang, X. Ding, G. Yuan, and X. Fu, "Dual-cameras-based driver's eye gaze tracking system with non-linear gaze point refinement," *Sensors*, vol. 22, no. 6, p. 2326, Mar. 2022.

[25] S. Senarath, P. Pathirana, D. Meedeniya, and S. Jayarathna, "Customer gaze estimation in retail using deep learning," *IEEE Access*, vol. 10, pp. 64904–64919, 2022.

[26] Y. Yin, H. Wang, S. Liu, J. Sun, P. Jing, and Y. Liu, "Internet of Things for diagnosis of Alzheimer's disease: A multimodal machine learning approach based on eye movement features," *IEEE Internet Things J.*, vol. 10, no. 13, pp. 11476–11485, Jul. 2023.

[27] N. Torok, V. Guillemin, and J. M. Barnothy, "LXXX photoelectric nystagmography," *Ann. Otol., Rhinol. Laryngol.*, vol. 60, no. 4, pp. 917–926, Dec. 1951.

[28] W. M. Smith and P. J. Warter, "Eye movement and stimulus movement; new photoelectric electromechanical system for recording and measuring tracking motions of the eye," *J. Opt. Soc. Amer.*, vol. 50, no. 3, pp. 245–250, 1960.

[29] C. Rashbass, "New method for recording eye movements," *J. Opt. Soc. Amer.*, vol. 50, no. 7, pp. 642–644, 1960.

[30] Y. Kong, S. Lee, J. Lee, and Y. Nam, "A head-mounted goggle-type video-oculography system for vestibular function testing," *EURASIP J. Image Video Process.*, vol. 2018, no. 1, p. 28, 2018.

[31] Y. Imaoka, A. Flury, and E. D. de Bruin, "Assessing saccadic eye movements with head-mounted display virtual reality technology," *Frontiers Psychiatry*, vol. 11, Sep. 2020, Art. no. 572938.

[32] O. H. Mowrer, T. C. Ruch, and N. E. Miller, "The corneo-retinal potential difference as the basis of the galvanometric method of recording eye movements," *Amer. J. Physiol.-Legacy Content*, vol. 114, no. 2, pp. 423–428, 1935.

[33] D. A. Robinson, "A method of measuring eye movemnent using a scieral search coil in a magnetic field," *IEEE Trans. Bio-Med. Electron.*, vol. BMEL-10, no. 4, pp. 137–145, Oct. 1963.

[34] E. G. Freedman and D. L. Sparks, "Coordination of the eyes and head: Movement kinematics," *Exp. Brain Res.*, vol. 131, no. 1, pp. 22–32, Mar. 2000.

[35] H. Wang, C. Pan, and C. Chaillou, "Tracking eye gaze under coordinated head rotations with an ordinary camera," in *Computer Vision—ACCV* (Lecture Notes in Computer Science), vol. 5995. Berlin, Germany: Springer, 2010, pp. 120–129.

[36] *Fixed Foveated Rendering (FFR)*. Oculus Documentation for Developers. Accessed: Jul. 6, 2023. [Online]. Available: https://developer.oculus.com/documentation/native/android/mobile-ffr/

[37] T. Okada, H. Yamazoe, I. Mitsugami, and Y. Yagi, "Preliminary analysis of gait changes that correspond to gaze directions," in *Proc. 2nd IAPR Asian Conf. Pattern Recognit.*, Nov. 2013, pp. 788–792.

[38] *HTC VIVE*. Accessed: Jul. 6, 2023. [Online]. Available: https://www.vive.com/

[39] *aGlass Eye Tracker*. Accessed: Jul. 6, 2023. [Online]. Available: https://xinreality.com/wiki/AGlass

[40] *HMMLearn*. Accessed: Jul. 6, 2023. [Online]. Available: https://github.com/hmmlearn/hmmlearn

[41] T. G. Dietterich, "Machine learning research: Four current directions," *Artif. Intell. Mag.*, vol. 18, no. 4, pp. 97–136, 1997.

[42] R. Polikar, "Ensemble based systems in decision making," *IEEE Circuits Syst. Mag.*, vol. 6, no. 3, pp. 21–45, 3rd Quart., 2006.

[43] L. Breiman, "Stacked regressions," *Mach. Learn.*, vol. 24, no. 1, pp. 49–64, Jul. 1996.

[44] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, Feb. 1970.

[45] R. Tibshirani, "Regression shrinkage and selection via the lasso: A retrospective," *J. Roy. Stat. Soc. Ser. B, Stat. Methodol.*, vol. 73, no. 3, pp. 273–282, Jun. 2011.

[46] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *Ann. Statist.*, vol. 32, no. 2, pp. 407–499, Apr. 2004.

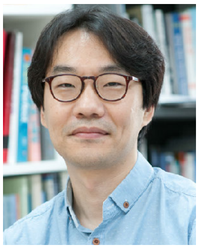
[47] E. Iturbide, J. Cerda, and M. Graff, "A comparison between LARS and LASSO for initialising the time-series forecasting auto-regressive equations," *Proc. Technol.*, vol. 7, pp. 282–288, Dec. 2013.

[48] Technical Blog of NVIDIA. *Easy VRS Integration With Eye Tracking*. Accessed: Jul. 6, 2023. [Online]. Available: https://developer.nvidia.com/blog/vrs-wrapper/

ARDIANTO SATRIAWAN received the B.S. and M.S. degrees in electrical engineering from the Bandung Institute of Technology (ITB), Indonesia. He is currently pursuing the Ph.D. degree in electrical and information engineering with the Seoul National University of Science and Technology, Seoul, South Korea. His current research interest includes virtual reality offloading.

AIRLANGGA ADI HERMAWAN received the B.S. degree in electrical engineering from Gadjah Mada University, Yogyakarta, Indonesia, and the M.S. degree in computer science and engineering from the Technical University of Eindhoven, Eindhoven, The Netherlands. He is currently pursuing the Ph.D. degree in electrical and information engineering with the Seoul National University of Science and Technology, Seoul, South Korea. His current research interest includes virtual reality offloading.

YAKUB FAHIM LUCKYARNO received the B.S. degree in physics from Gadjah Mada University, Yogyakarta, Indonesia, and the M.S. degree in computer engineering from the King Mongkut's Institute of Technology Ladkrabang, Bangkok. He is currently pursuing the Ph.D. degree in electrical and information engineering with the Seoul National University of Science and Technology, Seoul, South Korea. His current research interest includes virtual reality offloading.



JI-HOON YUN (Senior Member, IEEE) received the B.S. degree in electrical engineering and the M.S. and Ph.D. degrees in electrical engineering and computer science from Seoul National University (SNU), Seoul, South Korea, in 2000, 2002, and 2007, respectively.

He was a Postdoctoral Researcher with the Real-Time Computing Laboratory, The University of Michigan, Ann Arbor, MI, USA, in 2010; and a Senior Engineer with the Telecommunication Systems Division, Samsung Electronics, Suwon, South Korea, from 2007 to 2009. He is currently a Professor with the Department of Electrical and Information Engineering, Seoul National University of Science and Technology (SeoulTech), Seoul. Before joining SeoulTech, in 2012, he was with the Department of Computer Software Engineering, Kumoh National Institute of Technology, as an Assistant Professor. His current research interests include wireless communications, networking, and mobile applications.