3D Prior is All You Need: Cross-Task Few-shot 2D Gaze Estimation

Yihua Cheng[1], Hengfei Wang[1], Zhongqun Zhang[1], Yang Yue[1],
Bo Eun Kim[1,3], Feng Lu[2], Hyung Jin Chang[1]
[1]University of Birmingham, [2]Beihang University, [3]Dankook University

Abstract

3D and 2D gaze estimation share the fundamental objective of capturing eye movements but are traditionally treated as two distinct research domains. In this paper, we introduce a novel cross-task few-shot 2D gaze estimation approach, aiming to adapt a pre-trained 3D gaze estimation network for 2D gaze prediction on unseen devices using only a few training images. This task is highly challenging due to the domain gap between 3D and 2D gaze, unknown screen poses, and limited training data. To address these challenges, we propose a novel framework that bridges the gap between 3D and 2D gaze. Our framework contains a physics-based differentiable projection module with learnable parameters to model screen poses and project 3D gaze into 2D gaze. The framework is fully differentiable and can integrate into existing 3D gaze networks without modifying their original architecture. Additionally, we introduce a dynamic pseudo-labelling strategy for flipped images, which is particularly challenging for 2D labels due to unknown screen poses. To overcome this, we reverse the projection process by converting 2D labels to 3D space, where flipping is performed. Notably, this 3D space is not aligned with the camera coordinate system, so we learn a dynamic transformation matrix to compensate for this misalignment. We evaluate our method on MPIIGaze, EVE, and GazeCapture datasets, collected respectively on laptops, desktop computers, and mobile devices. The superior performance highlights the effectiveness of our approach, and demonstrates its strong potential for real-world applications.

1. Introduction

Gaze estimation tracks eye movements to predict human attention [13]. It is a highly applied research topic, where various application scenarios, such as intelligent vehicles [14, 22], VR/AR [25, 27, 30], and disease diagnosis [6, 33] demand distinct and specialized gaze estimation solutions.

Recent gaze estimation methods primarily focus on 3D gaze estimation [8, 35], wherein 3D direction vectors are derived from facial images. Such methods exhibit high adaptability, facilitating straightforward application in diverse environments [37]. However, they present limitations in practical applications, such as human-computer interaction, where precise gaze targets are essential. Existing approaches often require post-processing to calibrate the pose of interacted objects, e.g., a screen, and compute the intersection between gaze and objects [39]. This process poses significant challenges, particularly for non-expert users.

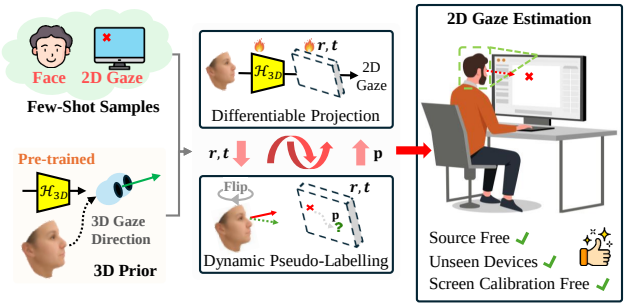

Figure 1. We introduce a novel cross-task few-shot 2D gaze estimation approach. Our method leverages a pre-trained 3D gaze estimation network and few-shot 2D gaze samples to achieve 2D gaze estimation on unseen devices. It contains a physics-based differentiable projection module to bridge 3D and 2D gaze, along with a dynamic pseudo-labelling strategy for 2D labels under unknown screen poses. Our approach is both screen-calibration-free and source-free, significantly expanding its application potential.

Conversely, some methods directly estimate 2D gaze within screen coordinate systems. Deep learning-based approaches utilize large training datasets to map facial images to 2D gaze [4, 23]. However, these models are often entangled with multiple device-specific factors. Traditional approaches construct 3D eye models using prior anatomical knowledge and fit these models with few-shot calibration images [17]. Although such methods require specialized equipment for precise eye tracking, they raise the question, *Can we achieve similar patterns within the deep learning paradigm for quick adaptation across various devices?*

In this work, we explore a novel topic, cross-task few-shot 2D gaze estimation. We observe that 3D gaze estimation has recently gained significant attention in the research community. It is performed within 3D space, free

from entanglement with specific devices. These insights suggest that 3D gaze estimation models would be a great prior, similar to the 3D anatomical eye model in traditional methods. Therefore, our approach aims to utilize 3D gaze estimation as prior and adapt it efficiently for 2D gaze estimation. However, this setting introduces several significant challenges, such as the domain gap between 3D and 2D gaze tasks, unseen device settings, i.e., *w/o* screen calibration, and insufficient training data. We show a comparison between our task with common methods in Table 1.

To address these challenges, we first propose a novel framework to bridge the gap between 2D and 3D gaze estimation. We decomposes 2D gaze estimation into two components: 3D gaze estimation and gaze projection. We first estimate 3D gaze from face images, and then project 3D gaze onto a specific 2D plane to infer the 2D gaze. Unlike existing methods that require screen calibration to obtain the screen pose [1, 13], our framework includes a physics-based differentiable projection module. This module models screen pose using six learnable parameters, i.e., rotation and translation vectors that map screen coordinate system to camera coordinate system. By implementing the projection in a fully differentiable manner, our framework enables seamless integration of the projection module into any existing 3D gaze estimation model without changing its original architecture. Furthermore, since the framework is fully differentiable, it supports fine-tuning on 2D annotated data.

We further propose a dynamic pseudo-labelling strategy for 2D label in our framework. Specifically, we perform flipping on face images and aim to assign pseudo-label for the flipped images. While this process is straightforward for 3D gaze annotations, it becomes more complex for 2D gaze due to dependencies on factors like head position and screen pose, especially the screen pose is unknown. To address this, we perform dynamic pseudo-labelling during training. In each iteration, we reverse the projection process using the learnable screen parameters to convert 2D labels into 3D labels. This allows us to perform flipping directly in the 3D gaze space. A key insight is that flipping needs to occur in the camera coordinate system, while accounting for a shift in coordinate systems during training. To handle this, we learn a dynamic transformation that maps the shifted system back to the camera coordinate system, ensuring reliable pseudo-label generation. Additionally, we apply color jittering during training, which does not alter the 2D gaze labels, and minimize uncertainty across jittered images to improve robustness.

Overall, our main contribution contains four-folds:

1. We explore the novel topic of cross-task few-shot 2D gaze estimation. This topic not only extends the application of 3D gaze research to the 2D domain but also provides a promising direction for real-world applications.
2. We propose a framework to bridge 3D and 2D gaze estimation, which includes a physics-based differentiable projection module with six learnable screen parameters to convert 3D gaze to 2D gaze. By leveraging this framework, we can quickly adapt a 3D gaze model for 2D gaze estimation using only a small number of images.
3. We propose a dynamic pseudo-labeling strategy for 2D labels in our framework. We reverse the projection using learnable screen parameters to convert 2D labels back into 3D labels and perform pseudo-labeling in the 3D gaze space. Furthermore, we learn a dynamic transformation to address the shifted coordinate system problem.
4. We establish a benchmark for the cross-task few-shot 2D gaze estimation, and evaluate our method on three datasets covering daily scenarios, including laptop, desktop computer and mobile devices. The superior performance demonstrates the advantage of our approach.

Table 1. Comparison between our method with existing methods. We introduce an unexplored task in gaze estimation, which aims to adapt 3D gaze models for 2D gaze estimation with few-shot data.

Category	Train	Test	Cross Env.	Cross Task	Methods
3D Gaze Estimation	**3D**	**3D**	×	×	[8, 12, 20]
2D Gaez Estiamtion	**2D**	**2D**	×	×	[4, 23]
Personalize	**2D**	**2D**	×	×	[18]
	3D	**3D**	✓	×	[24, 28]
Domain Adaption	**3D**	**3D**	✓	×	[2, 3, 7]
Ours	**3D**	**2D**	✓	✓	None

2. Related Works

2.1. Gaze Estimation

Gaze estimation methods are generally classified into 3D and 2D gaze estimation based on output [13]. 3D gaze estimation defines gaze as a directional vector originating from the face toward gaze targets [11, 35]. It typically focuses on enhancing accuracy and generalizability across diverse environments. Related research spans several fields, including supervised learning [8–10], unsupervised domain adaptation [2, 3, 7], feature disentanglement [12, 28, 34], etc.

On the other hand, 2D gaze estimation is primarily applied in screen-based contexts, where gaze is represented as a pixel coordinate in the screen coordinate system [4, 23]. Compared to 3D gaze estimation, 2D gaze estimation is more directly applicable to human-computer interaction [19, 26, 32]. However, it becomes entangled with multiple device-specific factors, such as screen size and camera-screen pose, which complicate generalization across various setups. The adaptation of 2D gaze estimation methods remains a notable research challenge.

Although these gaze estimation methods fundamentally capture eye movement, the distinct differences between 3D and 2D gaze annotations define them as separate research

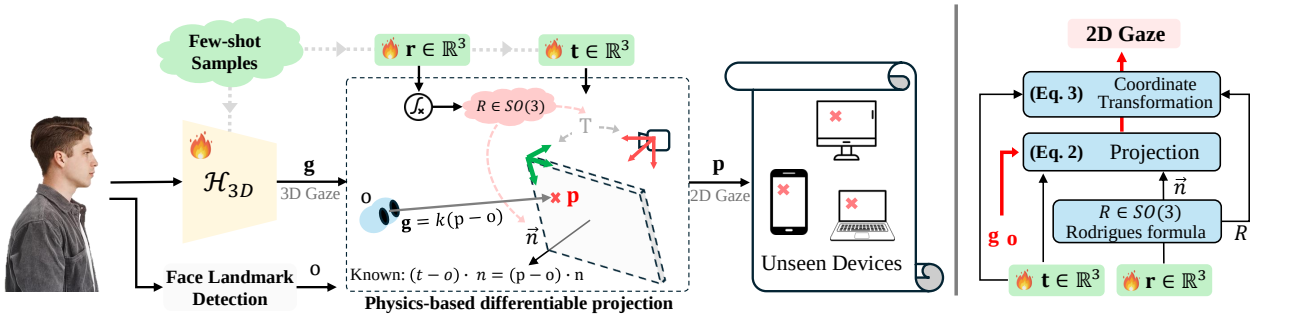

Figure 2. We propose a framework for the cross-task few-shot 2D gaze estimation. The framework contains a physics-based differentiable projection module with learnable parameters $\mathbf{r}$ and $\mathbf{t}$ to model screen, and project 3D gaze into 2D gaze. The framework is fully differentiable and can integrate into existing 3D gaze networks without modifying their original architecture. Leveraging this framework, we can quickly adapt a 3D gaze model for 2D gaze estimation using only a small number of images.

areas. In this work, we propose a framework to bridge the gap between 2D and 3D gaze estimation, enabling the direct application of 3D gaze research to 2D gaze estimation.

2.2. 2D Gaze Estimation via Projection

It is typical to compute the intersection between 3D gaze and a 2D plane for 2D gaze, a process referred to as gaze projection in this work. The most common application is in VR [15, 31], where the head-mounted display provides 3D gaze estimation, and developers can easily obtain a plane pose in VR space. They project the gaze onto this plane or determine if it intersects for interaction.

This strategy is also used in deep-learning based gaze estimation methods. They calibrate the screen pose during the post-processing stage and convert 3D gaze into 2D gaze on the screen [13, 39]. Recently, some methods have attempted to inject the projection into the deep learning framework. Balim et al. [1] first require screen calibration to obtain screen parameters and then model the projection process using the calibrated pose. Cheng et al. [14] focus on estimating gaze zones on vehicle windshields. They define a basis tri-plane, project 3D gaze onto this plane, and then learn a mapping from the interaction points to the gaze zone.

In our work, we model the full projection process by defining the screen pose with six learnable parameters. The projection module is parameter-efficient. More importantly, our method does not require screen calibration, which can be challenging for non-expert users.

3. Methodology

3.1. Task Definition

Given a pre-trained 3D gaze estimation network $\mathcal{H}_{3D}\left(\mathbf{I};\beta\right)$, which takes face images $\mathbf{I}$ as input and outputs 3D gaze direction $\mathbf{g}$, i.e., $\mathcal{H}_{3D}:\mathbf{I}\rightarrow\mathbf{g}$, our objective is to develop a 2D gaze estimation network $\mathcal{H}_{2D}\left(\mathbf{I};\theta\right)$. Using few-shot training samples $\mathcal{D}=\{(\mathbf{I}_i,\mathbf{p}_i)\}_{i=1}^{N}$, where N is the number of training samples, this network estimates 2D pixel coordinates $\mathbf{p}$ from face images, i.e., $\mathcal{H}_{2D}:\mathbf{I}\rightarrow\mathbf{p}$. We consider a restricted setting where: 1) the method is source-free, as the training set of $\mathcal{H}_{3D}$ is unavailable, and 2) it is screen calibration-free, with the screen pose unspecified. These restrictions make our method convenient for practical applications while upholding data privacy.

3.2. Physics-Based Differentiable Projection

Our work aims to learn a new $\mathcal{H}_{2D}$ using few-shot samples. The primary challenge lies in transferring knowledge from $\mathcal{H}_{3D}$ to $\mathcal{H}_{2D}$. However, the two networks perform different tasks, making some conventional methods such as feature distillation unsuitable. To address this, our idea is to decompose the 2D gaze estimation into 3D gaze estimation and gaze projection. Specifically, we incorporate $\mathcal{H}_{3D}$ as part of $\mathcal{H}_{2D}$, supplemented with an additional module for projecting gaze directions onto a 2D screen. Unlike existing gaze projection strategies that often rely on post-processing [13] or require screen calibration [1], we introduce a physics-based differentiable projection module. This module models screen pose as learnable weights, enabling the projection process to occur in a differentiable and adaptable manner.

In detail, we define learnable weights $\mathbf{r}\in\mathbb{R}^3$ as the rotation vector and $\mathbf{t}\in\mathbb{R}^3$ as the translation vector within the projection module, establishing the transformation from the screen coordinate system to the camera coordinate system. To transform $\mathbf{r}$ into the rotation matrix $\mathbf{R}\in\mathbb{R}^{3\times3}$, we apply the Rodrigues formula, which preserves the orthogonality of $\mathbf{R}$ so that $\mathbf{R}\in SO(3)$. The input of projection module contains gaze direction $\mathbf{g}\in\mathbb{R}^3$ and the 3D position of the face center $\mathbf{o}\in\mathbb{R}^3$, the latter of which can be computed using existing 3D landmark estimation methods [16]. Overall, the module $\mathcal{P}$ could be denoted as:

$$\hat{\mathbf{p}}=\mathcal{P}(\mathbf{g},\mathbf{o};\mathbf{r},\mathbf{t}), \tag{1}$$

where $\hat{\mathbf{p}}\in\mathbb{R}^2$ represents the estimated screen coordinate.

We first compute the intersection points between gaze directions and the learnable screen denoted with $(\mathbf{r},\mathbf{t})$. To

establish the screen pose, we need a normal vector $\mathbf{n}$ and a point coordinate on the screen. The normal vector can be derived using $\mathbf{R}[:, 2]$, i.e., the third column of the rotation matrix [13], while $\mathbf{t}$ serves as a reference point on the plane. Given that the dot product between the normal vector of a plane and the vector connecting any point on the plane to a fixed point is constant, it is obvious that the intersection point $\mathbf{p}_{3D}$ is

$$\mathbf{p}_{3D} = \mathbf{o} + \frac{(\mathbf{t} - \mathbf{o}) \cdot \mathbf{n}}{\mathbf{g} \cdot \mathbf{n}} \mathbf{g}, \tag{2}$$

Note that $\mathbf{p}_{3D}$ represents coordinates in the camera coordinate system. To convert it to the screen coordinate system, we apply

$$\mathbf{p} = \mathbf{R}^{-1}(\mathbf{p}_{3D} - \mathbf{t}), \tag{3}$$

We slightly abuse the notation $\mathbf{p}$ in Eq. 3, where the final 2D gaze coordinate corresponds to the first two components of $\mathbf{p}$. These values can then be further converted into pixel coordinates by utilizing the screen's PPI (pixels per inch), which is easily obtainable as a screen parameter.

Therefore, the network $\mathcal{H}_{2D}$ can be denoted as

$$\mathcal{H}_{2D}(\mathbf{I}, \mathbf{o}; \beta, \mathbf{r}, \mathbf{t}) = \mathcal{P}(\mathcal{H}_{3D}(\mathbf{I}), \mathbf{o}), \tag{4}$$

and the objective function is denoted as

$$\min_{\beta, \mathbf{r}, \mathbf{t}} \sum_{i=1}^{N} \|\mathcal{H}_{2D}(\mathbf{I}_i, \mathbf{o}_i) - \mathbf{p}_i\|_1 \tag{5}$$

We illustrate the projection module in Figure 2.

3.3. Dynamic Pseudo-Labeling for 2D Gaze

Data augmentation is a typical technique to improve model performance, particularly with limited dataset sizes. In this section, we apply flipping to expand the data space. In 3D gaze estimation, the flipping involves horizontally flipping face image and adjusting the label by negating the x-coordinate value. We formally define the operation in label as $\mathcal{F}(\mathbf{g})$. However, generating reliable pseudo labels after flipping is challenging for 2D gaze estimation.

Our core idea is to dynamically generate pseudo-labels during training by leveraging the differentiable projection module within our framework, which includes learnable screen parameters. This enables us to address the challenges of assigning 2D pseudo-labels by reversing the projection process, i.e., converting 2D screen coordinates into 3D gaze directions, where we can then apply flipping in 3D space. The pseudo-labeling function $\mathcal{Q}(\mathbf{p})$ is defined as

$$\mathcal{Q}(\mathbf{p}) = \mathcal{P}(\mathcal{F}(\mathcal{P}^{-1}(\mathbf{p}))), \tag{6}$$

where $\mathcal{P}^{-1}$ represents the reverse projection process. Specifically, we first transform $\mathbf{p}$ into the camera coordinate system. The gaze direction is then defined as the vector originating from the face center and directed toward the gaze point,

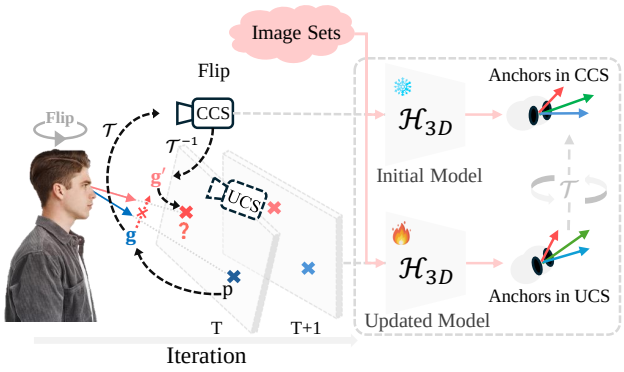

Figure 3. The dynamic pseudo-labeling strategy for 2D gaze involves reversing the projection process to convert 2D gaze into 3D space, where we compute pseudo-labels. To align the camera coordinate system (CCS) with the unknown coordinate system (UCS), we use the same image sets as input to both the initial and the updated 3D model. The initial model, trained on the CCS, while the updated model operates within the UCS. By leveraging the outputs from these models as two anchors, we derive the transformation $\mathcal{T}$ to align the coordinate systems. Notably, $\mathcal{T}$ should be invertible.

$$\mathcal{P}^{-1}(\mathbf{p}, \mathbf{o}) = (\mathbf{R}\mathbf{p} + \mathbf{t}) - \mathbf{o}, \tag{7}$$

which we normalize to ensure the vector has a unit length.

However, we observed that assigning pseudo-labels as Eq. 6 led to model collapse, with the pseudo-labels diverging to large values during training. On the other hand, we found that $\mathcal{H}_{2D}$ struggled to learn the correct screen parameters, and noted substantial changes in the 3D gaze estimation network itself. Our intuition suggests that *changing the screen pose should theoretically allow us to find an optimal screen pose, but this could also be approached by rotating the camera instead, i.e., optimize* $\mathcal{H}_{3D}$.

Based on the observations, we find that Eq. 6 is not consistently reliable. The key insight is that human gaze direction is inherently defined in the camera coordinate system. Flipping image affects the camera coordinate system itself, meaning the gaze label should be adjusted accordingly, i.e., *flipping should be performed in camera coordinate system*. However, since the 3D gaze estimation network undergoes updates during fine-tuning, it is shifted into an unknown coordinate system. This change in coordinate systems disrupts the alignment of gaze labels, leading to model collapse.

To solve this problem, we aim to learn a transformation $\mathcal{T}$ that maps the unknown coordinate system to camera coordinate system. Our idea is to identify anchors in the two coordinate systems, allowing us to model this problem as an alignment task. Specifically, we denote the initial pre-trained 3D gaze estimation network as $\mathcal{H}_{3D}(;\beta_0)$ and the fine-tuned network as $\mathcal{H}_{3D}(;\beta_k)$. Notably, $\mathcal{H}_{3D}(;\beta_0)$ is pre-trained in the camera coordinate system, while $\mathcal{H}_{3D}(;\beta_k)$ operates in the unknown coor-

dinate system. Therefore, we can acquire the anchor as $\{\mathcal{H}_{3D}(\mathbf{I}_i;\beta_0)\}_{i=1}^N$ and $\{\mathcal{H}_{3D}(\mathbf{I}_i;\beta_k)\}_{i=1}^N$ using training set $\mathcal{D}$. The alignment problem can then be formulated as:

$$\min_{\mathcal{T}} \sum_{i=1}^{N} \|\mathcal{T}\mathcal{H}_{3D}(\mathbf{I}_i;\beta_k) - \mathcal{H}_{3D}(\mathbf{I}_i;\beta_0)\|_2 \tag{8}$$

Notably, $\mathcal{T}$ should be invertible. Therefore, we model the transformation as a rotation operation, enabling us to solve it using singular value decomposition (SVD). We have

$$[U,S,V] = \text{SVD}(\mathcal{H}_{3D}(\mathbf{I}_i;\beta_i) * \mathcal{H}_{3D}(\mathbf{I}_i;\beta_0)^T), \tag{9}$$

and $\mathcal{T} = VU^T$. Consequently, we can update Eq. 6 as follows

$$\mathcal{Q}(\mathbf{p}) = \mathcal{P}(\mathcal{T}^{-1} * \mathcal{F}(\mathcal{T} * \mathcal{P}^{-1}(\mathbf{p}))), \tag{10}$$

where $*$ represents matrix multiplication. $\mathcal{Q}(\mathbf{p})$ is also dynamic and re-computed during each iteration since the coordinate system continues to change throughout fine-tuning.

The objective function is denoted as

$$\min_{\beta,\mathbf{r},\mathbf{t}} \sum_{i=1}^{N} \|\mathcal{H}_{2D}(\mathbf{I}'_i,\mathbf{o}_i) - \mathcal{Q}(\mathbf{p}_i)\|_1 \tag{11}$$

Where $\mathbf{I}'_i$ is the flipping image of $\mathbf{I}_i$.

3.4. Minimize Uncertainty across Jittered Images

We also perform color jitter and minimize uncertainty across jittered images to enhance model robustness. Given a face image $\mathbf{I}$, we apply color jitter $\mathcal{J}$ to create a set of augmented images, $\{\mathcal{J}_k(\mathbf{I})\}_{k=1}^K$, where k represents the number of random color jitters performed. We minimize the variance in the gaze predictions for this set. Specifically, we pass each augmented image through the model, obtaining predictions $\{\mathcal{H}_{2D}(\mathcal{J}_k(\mathbf{I}))\}_{k=1}^K$. We calculate the centroid of these predictions and minimize the distance between each prediction and the centroid. Additionally, we also minimize the distance between the predictions and the ground truth. To stabilize training, we introduce a temporal weight $\tau = \frac{t-1}{t}$ for the variance loss, starting with a smaller weight that increases over epoch t. The loss is defined as

$$\begin{aligned}\mathcal{L}_{unc} = &\frac{1}{NK}\sum_{i=1}^{N}\sum_{k=1}^{K}(\|\mathcal{H}_{2D}(\mathcal{J}_k(\mathbf{I}_i)) - \mathbf{p}_i\|_1 + \\ &\tau * \|\mathcal{H}_{2D}(\mathcal{J}_k(\mathbf{I}_i)) - \frac{1}{K}\sum_{j=1}^{K}\mathcal{H}_{2D}(\mathcal{J}_j(\mathbf{I}_i))\|_2)\end{aligned} \tag{12}$$

The temporal weight mitigates the risk of model collapse, as we observe that the second term of $\mathcal{L}_{unc}$ tends to be large at the start of training, and a high initial learning weight can lead to instability. Additionally, we apply L2 regularization to the second term since it assigns greater weight to outliers.

3.5. Implementation Details

Our model is optimized using the loss functions defined in Eq. 5, Eq. 11 and Eq. 12, with corresponding weights of 1, 0.4, and 0.25, respectively. For training, we set $N = 10$, meaning the training set contains 10 samples, and $K = 4$, meaning we apply four random color jitter augmentations per iteration. The model is implemented in PyTorch and trained on an NVIDIA RTX 3090. We train for 80 epochs, setting the learning rate initially to 0.001, with a 5-epoch warmup phase. After 60 epochs, the learning rate decays to 0.0005. We use GazeTR [8] (ResNet18 + 6-layer transformer) pretrained on Gaze360 [21] as the basic 3D model. Please refer the supplementary material for more details.

4. Experiment

4.1. Setup

In this paper, we propose a cross-task few-shot 2D gaze estimation task. We first build the evaluation benchmark.

Datasets: We evaluate methods on three datasets: MPIIGaze [36], EVE [29], and GazeCapture [23]. These datasets were collected in different devices, including laptops, desktop computers, and mobile devices. By assessing performance across these datasets, we demonstrate the generalization capability of methods across various devices.

Data Preprocessing: Image normalization [13] is usually used to enhance 3D gaze estimation performance. In our work, we utilize the normalized images provided by the MPIIGaze and EVE datasets, and implement the method [38] for normalizing the GazeCapture. Note that, the normalization changes 3D gaze with a rotation matrix. Although our work does not use the 3D label, the predicted 3D gaze should be transformed back for projection. Furthermore, the MPIIGaze dataset augments 3D gaze estimation data by flipping images, which is not applicable for 2D gaze estimation. We exclude the flipped images for consistency. The EVE dataset provides videos along with corresponding gaze trajectories. We sample one frame for every 20 frames to construct the benchmark. We sample 20 subjects in GazeCapture dataset, ensuring that each has at least 500 images. We clean the dataset to remove images without face. Notably, four of the 20 subjects used a tablet for data collection, while the rest used phones. Please refer the supplementary materials for more details.

Evaluation Metric: We perform person-specific evaluation and report the average performance across subjects for comparison. Performance is measured as the Euclidean distance (in mm) between predictions and ground truth, where lower values indicate better accuracy.

4.2. Quantitative Comparison

We first compare our method with existing approaches EFE [1] and IVGaze [14]. EFE is an end-to-end gaze esti-

Table 2. Quantitative evaluation. Our method achieves best result among comparison methods. We also report the performance of 2D gaze estimation methods in the second row for reference.

Method	Training Samples	EVE [29]	MPIIGaze [35]	GazeCapture [23]
iTracker [23]	All dataset	-	-	26.8
EyeNet [29]		49.7	-	-
Full-Face [36]		38.6	42.0	-
AFF-Net [4]		-	39.0	19.6
EFE [1]		38.5	38.9	20.5
EFE [1]	10	64.9 ▼33%	100.2 ▼43%	48.5 ▼26%
IVGaze [14]	10	177.7 ▼75%	132.2 ▼57%	68.1 ▼47%
Ours	10	43.4	56.7	35.7

mation method that includes a projection module to convert 3D gaze predictions into 2D gaze. IVGaze utilizes a basis tri-plane for projection, followed by a lightweight transformer to refine the projection points. For a fair comparison, we re-implement both methods using the same 3D gaze estimation network and pre-trained weights as our method. Our goal is to evaluate the performance differences resulting from different projection strategies. Notably, EFE requires screen calibration for the projection; to ensure fairness, we set these screen parameters as learnable and initialize them with the same values used in our method. The results of these comparisons are presented in Table 2.

IVGaze includes a transformer to refine projection points. While this transformer performs well when trained on the full dataset, it struggles with limited data, leading to underfitting when trained on just 10 samples. This results in poor performance on the EVE and MPIIGaze datasets, highlighting the advantage of our approach. In contrast, our method avoids the use of complex architectures that can suffer from underfitting in few-shot learning tasks. Instead, we directly model the projection process, leading to superior performance. On the other hand, EFE demonstrates reasonable performance, but our method achieves over 25% improvement across all three datasets. This significant boost is attributed to our more comprehensive modelling of the projection process, which reduces fitting complexity and naturally enhances overall performance.

We also report the performance of 2D gaze estimation methods trained on the entire dataset for reference. Note that they are not directly comparable to our method since both the training and test sets differ. These results are summarized in the second row of Figure 2. Our method achieves similar performance using only 10 images.

4.3. Comparison with Different Adaption Strategy

In this section, we evaluate the accuracy of different adaption strategies for obtaining 2D gaze from 3D predictions.

Direct Projection: We directly project the 3D gaze predictions from our pre-trained 3D gaze estimation network onto the screen using the known screen pose, providing a baseline performance measure for the network. This is not performed on GazeCapture, as it lacks reliable screen pose.

Direct Learning: We retain the architecture of the 3D gaze network and directly fine-tune it using the 2D annotations. Additionally, we concatenate the gaze origin $\mathbf{o}$ with the predicted gaze and use a MLP to map them to 2D gaze predictions. We then fine-tune this extended network and report the performance in Direct Learning (with $\mathbf{o}$).

Learning with Known Pose: Our method assumes the screen pose is unavailable. In this strategy, we change the learnable parameters as the ground truth screen pose.

The result is shown in Table 3. The Direct Projection method struggles to perform effectively on the EVE and MPIIGaze datasets without fine-tuning. However, integrating it into our framework yields over 40% improvement, demonstrating the critical role of our learning framework. The Direct Learning strategy, on the other hand, fails to achieve reasonable performance due to the substantial domain gap between 3D and 2D gaze estimation. We compare its performance with Direct Projection. The learning strategy does not show any performance gains, which highlights the challenge of adapting 3D gaze models to 2D tasks. Even when the gaze origin is included as an additional feature, the limited training data makes it challenging for the model to learn the complex mapping. In contrast, our framework leverages physics-based differentiable projection, enabling it to achieve superior performance. The Learning with Known Pose method outperforms our method due to access to the known screen pose, highlighting the importance of accurate screen pose information for 2D gaze estimation.

Table 3. Comparison with different 3D to 2D adaption strategy. We direct project 3D gaze to 2D gaze using the known screen pose without fine-tuning, which shows the advantage of our learning framework. We directly learn 2D gaze from 3D gaze with MLP, which highlights the challenges in the adaption from 3D model to 2D gaze estimation. We also show the performance when the learnable parameters is set as known pose in our method.

Strategy	EVE	MPIIGaze	GazeCapture
Direct Projection	80.5	101.9	N/A
Direct Learning	180.6	133.9	74.23
Direct Learning (with $\mathbf{o}$)	116.6	108.2	149.7
Learning with Known Pose	39.4	56.6	N/A
Ours	43.4	56.7	35.7

Table 4. We perform an ablation study to evaluate the impact of the dynamic pseudo-labeling strategy (PS-Label) and the loss to minimize uncertainty across jittered images ($\mathcal{L}_{unc}$). Both the two modules contribute to performance improvements.

Proj.	PS-Label	$\mathcal{L}_{unc}$	EVE	MPIIGaze	GazeCapture
✓			46.6	60.3	36.8
✓	✓		45.3	57.9	35.7
✓	✓	✓	43.4	56.7	35.7

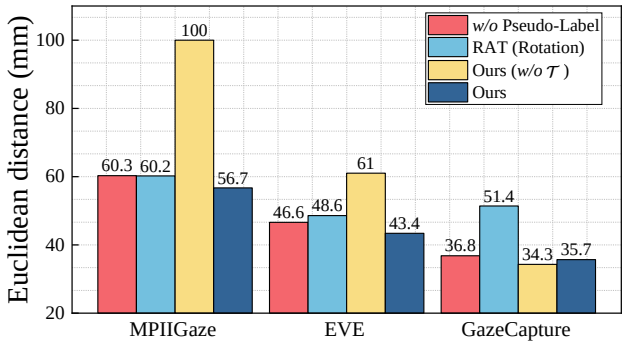

Figure 4. We compare the performance across different pseudo-labelling strategies. The red bar represents the projection without pseudo-labelling, serving as a baseline for comparison. We implemented RAT [5], which show no performance improvement over the baseline. Additionally, we evaluated our method without the transformation $\mathcal{T}$. In this case, the unreliable pseudo-labels resulted in a significant performance drop on the MPIIGaze and EVE datasets. Interestingly, omitting $\mathcal{T}$ led to improved results on the GazeCapture dataset. We found that this was because the initial screen pose happened to be same as the actual screen pose.

4.4. Ablation Study

We perform an ablation study to demonstrate the contribution of each module in our work. We first evaluate the performance when only the projection module is added to the pre-trained 3D gaze estimation network and fine-tuned. The results are shown as *Proj.* in Table 4. Compared to the results in Table 3, the projection module provides a significant performance improvement as it explicitly modelling the projection process, which effectively bridges the gap between 3D and 2D gaze estimation. Next, we introduce our dynamic pseudo-labeling strategy and minimize the uncertainty across jittered images. Both mechanisms bring performance improvements across all datasets.

The dynamic pseudo-labeling strategy is a key contribution of our work. To better understand its impact, we conduct a detailed comparison, as shown in Figure 4. We perform an ablation on the learning transformation $\mathcal{T}$ in our strategy. The results show a significant performance drop on the MPIIGaze and EVE datasets without $\mathcal{T}$, as unreliable pseudo-labels can cause model collapse during learning, especially with small training dataset sizes. Interestingly, we observe improved performance on the GazeCapture dataset without using $\mathcal{T}$. The authors of GazeCapture create a unified prediction space for 2D gaze, centered at the phone camera position. Our model initializes the screen pose as $\mathbf{t} = (0, 0, 0)$, making the initial pose closely approximate the real one. However, it is important to note that such cases are uncommon in real-world scenarios. Our method first converts 2D gaze to 3D space and learns $\mathcal{T}$ to align this space with the camera coordinate system. When the screen pose aligns exactly with ground truth, the 3D space already corresponds to the camera coordinate system. To establish the alignment, we use the predictions from the 3D gaze network as anchors for the camera coordinate system, which may introduce some bias. Nonetheless, our method demonstrates performance improvements compared to methods without pseudo-labeling.

We also implement existing method RAT [5], which assigns pseudo-label for rotated images. We convert 2D gaze into 3D gaze using learnable screen parameters, and perform RAT to augment training. RAT cannot bring performance improvement compared with the baseline.

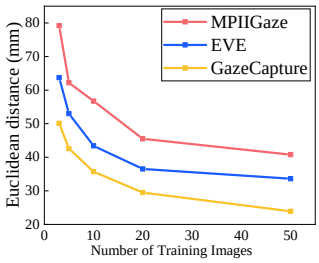

Figure 5. Performance with different number of training images.

#Training Images	Speed (sec/epoch)
3	0.89
5	0.90
10	0.91
20	0.96
50	1.16

Table 5. The model training time with different number of training samples.

4.5. Different Numbers of Training Images

In this section, we evaluate the effect of the number of training images on model performance. We experiment with different numbers of training images set to 3, 5, 10, 20, and 50, respectively. The performance is assessed across all three datasets, with results depicted in Figure 5. As shown, increasing the number of training images consistently improves the model performance.

Additionally, we measure the model training time when using varying numbers of training images, as summarized in Table 5. On average, each epoch takes approximately 0.9 seconds to process. Since our method does not require a large dataset, all images can be efficiently processed within a single epoch. With a total of 80 epochs, the complete training time is approximately 1.2 minute. Notably, this timing was tested in a Python environment and could be further optimized to achieve even faster performance with specific optimizations. This demonstrates significant real-time application potential for our method.

4.6. Repeatability Experiment

In this section, we conduct a robustness evaluation by training our method 10 times using different training samples in MPIIGaze to assess the impact of sample variability on model performance. We evaluate the performance on all 15 subjects for each trial and report the performance distribution. The results are visualized in a boxplot in Figure 6. The horizontal axis represents each of the 10 trials and each trial contains performance of 15 subjects. The box

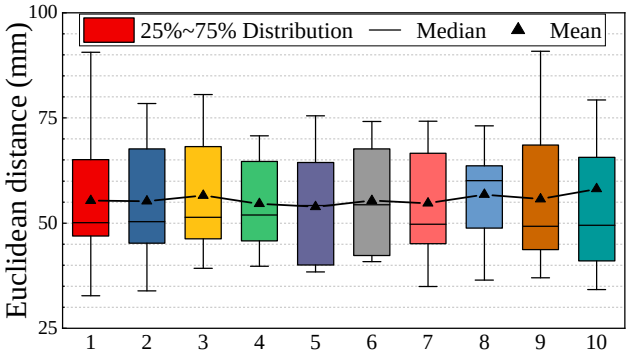

Figure 6. We train our method 10 times using different image samples in MPIIGaze for robustness evaluation. The horizontal axis corresponds to each of the 10 trials, while each bar shows the accuracy distribution across 15 subjects. The box depicts the interquartile range (25% to 75%), while the error bars covers the entire accuracy distribution. The average accuracy across 10 trials is 55.6, demonstrating the stability and robustness of our method.

depicts the interquartile range (25% to 75%), while the error bars cover the entire performance distribution. The triangle symbol indicates the average performance, and the black line represents the median performance. The average performance across all 10 trials is 55.6, which is slightly better than our previously reported value of 56.7. These results demonstrate the stability and robustness of our method despite variations in the training samples.

4.7. The Trajectories of Pseudo-Label

Our method contains a dynamic pseudo-labeling strategy to assign pseudo 2D labels for flipped images. To gain deeper insights into this process, we visualize the trajectories of the pseudo-labels over the course of 80 epochs in Figure 7. In addition, we compare the effect of our transformation strategy by plotting the pseudo-label positions without the transformation, i.e., the difference between Eq. 6 and Eq. 10. Both approaches share the same initial pseudo-labels. For reference, we also compute the ground truth labels using the calibrated screen pose for flipped images.

As shown in Figure 7, the initial pseudo-labels have a significant offset from the ground truth. However, our method dynamically updates the pseudo-labels based on the fine-tuned network, progressively aligning them closer to the ground truth with each iteration. By the end of training, the pseudo-labels have only minimal offsets from the ground truth, demonstrating the effectiveness of our approach. In contrast, the strategy without transformation fails to produce reliable pseudo-labels, leading to consistently large offsets from the ground truth. This comparison highlights the importance of our dynamic transformation mechanism in improving labelling accuracy.

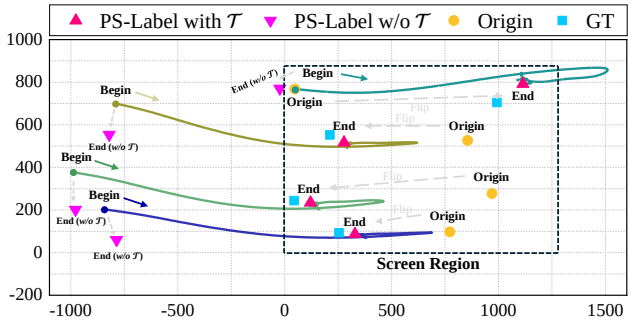

Figure 7. We visualize four trajectories of the pseudo-labels in our dynamic pseudo-labeling strategy. The ground truth for flipped images is computed using known screen pose. It is evident that our method progressively aligns the pseudo-labels closer to the ground truth. Additionally, we plot the pseudo-labels without applying $\mathcal{T}$ in our strategy, which shows a failure to produce reliable pseudo-labels, resulting in significant deviations from the ground truth.

5. Conclusion and Discussion

In this work, we introduce a novel cross-task few-shot 2D gaze estimation method. By leveraging few-shot 2D samples, we adapt a 3D gaze model to 2D gaze estimation on unseen devices. Since the 3D gaze network is trained in 3D space without being tied to specific devices, it theoretically maintains robust performance across different platforms. Our experiments validate this by proving results on three datasets. Besides, the adaption is rapid and source-free, significantly broadening its practical applicability.

Limitation: Our method infers 2D gaze through mathematical derivation within the differentiable projection module. While this approach enhances model interpretability and reliability, it can occasionally result in failure cases. For instance, when the input images lack visible faces, the predicted 3D gaze can become erratic. In such scenarios, the intersection point between the 3D gaze vector and the screen plane may significantly deviate from the ground truth. This issue arises because, unlike neural networks that constrain outputs to a plausible range, a purely mathematical projection may yield extreme values, e.g., when the 3D gaze is nearly parallel to the plane. Although these cases can be easily flagged in real-world applications, they may introduce biases during evaluation.

Future Directions: In this paper, we address the challenge of 2D pseudo-labeling. However, several open questions remain. For instance, can we leverage unlabeled face images to further enhance performance? Traditional methods often utilize a standard calibration pattern, could we incorporate a similar strategy? It is worth noting that our approach requires collecting samples initially, akin to a calibration process. We argue that this step is essential as it provides the necessary anchors for adapting to unseen devices. Nonetheless, exploring user-unaware calibration techniques is also a promising direction for future research.

References

[1] Haldun Balim, Seonwook Park, Xi Wang, Xucong Zhang, and Otmar Hilliges. Efe: End-to-end frame-to-gaze estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 2688–2697, 2023. 2, 3, 5, 6

[2] Yiwei Bao and Feng Lu. From feature to gaze: A generalizable replacement of linear layer for gaze estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1409–1418, 2024. 2

[3] Yiwei Bao and Feng Lu. Unsupervised gaze representation learning from multi-view face images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1419–1428, 2024. 2

[4] Yiwei Bao, Yihua Cheng, Yunfei Liu, and Feng Lu. Adaptive feature fusion network for gaze tracking in mobile tablets. In *International Conference on Pattern Recognition (ICPR)*, 2020. 1, 2, 6

[5] Yiwei Bao, Yunfei Liu, Haofei Wang, and Feng Lu. Generalizing gaze estimation with rotation consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4207–4216, 2022. 7

[6] Moinak Bhattacharya, Shubham Jain, and Prateek Prasanna. Gazeradar: A gaze and radiomics-guided disease localization framework. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 686–696. Springer, 2022. 1

[7] Xin Cai, Jiabei Zeng, Shiguang Shan, and Xilin Chen. Source-free adaptive gaze estimation by uncertainty reduction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22035–22045, 2023. 2

[8] Yihua Cheng and Feng Lu. Gaze estimation using transformer. *ICPR*, 2022. 1, 2, 5

[9] Yihua Cheng and Feng Lu. Dvgaze: Dual-view gaze estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 20632–20641, 2023.

[10] Yihua Cheng, Shiyao Huang, Fei Wang, Chen Qian, and Feng Lu. A coarse-to-fine adaptive network for appearance-based gaze estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020. 2

[11] Yihua Cheng, Xucong Zhang, Feng Lu, and Yoichi Sato. Gaze estimation by exploring two-eye asymmetry. *IEEE Transactions on Image Processing*, 29:5259–5272, 2020. 2

[12] Yihua Cheng, Yiwei Bao, and Feng Lu. Puregaze: Purifying gaze feature for generalizable gaze estimation. *AAAI*, 2022. 2

[13] Yihua Cheng, Haofei Wang, Yiwei Bao, and Feng Lu. Appearance-based gaze estimation with deep learning: A review and benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024. 1, 2, 3, 4, 5

[14] Yihua Cheng, Yaning Zhu, Zongji Wang, Hongquan Hao, Yongwei Liu, Shiqing Cheng, Xi Wang, and Hyung Jin Chang. What do you see in vehicle? comprehensive vision solution for in-vehicle gaze estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 1, 3, 5, 6

[15] Brendan David-John, Candace Peacock, Ting Zhang, T. Scott Murdison, Hrvoje Benko, and Tanya R. Jonker. Towards gaze-based prediction of the intent to interact in virtual reality. In *ACM Symposium on Eye Tracking Research and Applications*, 2021. 3

[16] Jianzhu Guo, Xiangyu Zhu, Yang Yang, Fan Yang, Zhen Lei, and Stan Z Li. Towards fast, accurate and stable 3d dense face alignment. In *The European Conference on Computer Vision*, 2020. 3

[17] Dan Witzner Hansen and Qiang Ji. In the eye of the beholder: A survey of models for eyes and gaze. *IEEE transactions on pattern analysis and machine intelligence*, 32(3):478–500, 2009. 1

[18] Zhe He, Adrian Spurr, Xucong Zhang, and Otmar Hilliges. Photo-realistic monocular gaze redirection using generative adversarial networks. In *The IEEE International Conference on Computer Vision*, 2019. 2

[19] Sinh Huynh, Rajesh Krishna Balan, and JeongGil Ko. imon: Appearance-based gaze tracking system on mobile devices. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 5 (4), 2022. 2

[20] Swati Jindal, Mohit Yadav, and Roberto Manduchi. Spatio-temporal attention and gaussian processes for personalized video gaze estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 604–614, 2024. 2

[21] Petr Kellnhofer, Adria Recasens, Simon Stent, Wojciech Matusik, and Antonio Torralba. Gaze360: Physically unconstrained gaze estimation in the wild. In *The IEEE International Conference on Computer Vision*, 2019. 5

[22] Muhammad Qasim Khan and Sukhan Lee. Gaze and eye tracking: Techniques and applications in adas. *Sensors*, 19 (24), 2019. 1

[23] Kyle Krafka, Aditya Khosla, Petr Kellnhofer, Harini Kannan, Suchendra Bhandarkar, Wojciech Matusik, and Antonio Torralba. Eye tracking for everyone. In *The IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 1, 2, 5, 6

[24] Erik Lindén, Jonas Sjöstrand, and Alexandre Proutiere. Learning to personalize in appearance-based gaze tracking. In *The IEEE International Conference on Computer Vision Workshops*, pages 1140–1148, 2019. 2

[25] Katerina Mania, Ann McNamara, and Andreas Polychronakis. Gaze-aware displays and interaction. In *ACM SIGGRAPH 2021 Courses*, pages 1–67, 2021. 1

[26] Omar Namnakani, Penpicha Sinrattanavong, Yasmeen Abdrabou, Andreas Bulling, Florian Alt, and Mohamed Khamis. Gazecast: Using mobile devices to allow gaze-based interaction on public displays. In *Proceedings of the 2023 Symposium on Eye Tracking Research and Applications*, New York, NY, USA, 2023. Association for Computing Machinery. 2

[27] A. Palazzi, D. Abati, s. Calderara, F. Solera, and R. Cucchiara. Predicting the driver's focus of attention: The dr(eye)ve project. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(7):1720–1733, 2019. 1

[28] Seonwook Park, Shalini De Mello, Pavlo Molchanov, Umar Iqbal, Otmar Hilliges, and Jan Kautz. Few-shot adaptive gaze estimation. In *The IEEE International Conference on Computer Vision*, 2019. 2

[29] Seonwook Park, Emre Aksan, Xucong Zhang, and Otmar Hilliges. Towards end-to-end video-based eye-tracking. In *The European Conference on Computer Vision*, pages 747–763. Springer, 2020. 5, 6

[30] Anjul Patney, Marco Salvi, Joohwan Kim, Anton Kaplanyan, Chris Wyman, Nir Benty, David Luebke, and Aaron Lefohn. Towards foveated rendering for gaze-tracked virtual reality. *ACM Transactions on Graphics (TOG)*, 35(6):1–12, 2016. 1

[31] Thammathip Piumsomboon, Gun Lee, Robert W. Lindeman, and Mark Billinghurst. Exploring natural eye-gaze-based interaction for immersive virtual reality. In *2017 IEEE Symposium on 3D User Interfaces (3DUI)*, pages 36–39, 2017. 3

[32] Nachiappan Valliappan, Na Dai, Ethan Steinberg, Junfeng He, Kantwon Rogers, Venky Ramachandran, Pingmei Xu, Mina Shojaeizadeh, Li Guo, Kai Kohlhoff, et al. Accelerating eye movement research via accurate and affordable smartphone eye tracking. *Nature communications*, 11(1): 4553, 2020. 2

[33] Sheng Wang, Xi Ouyang, Tianming Liu, Qian Wang, and Dinggang Shen. Follow my eye: Using gaze to supervise computer-aided diagnosis. *IEEE Transactions on Medical Imaging*, 41(7):1688–1698, 2022. 1

[34] Pengwei Yin, Jingjing Wang, Guanzhong Zeng, Di Xie, and Jiang Zhu. Lg-gaze: Learning geometry-aware continuous prompts for language-guided gaze estimation. In *The European Conference on Computer Vision*, 2024. 2

[35] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. Appearance-based gaze estimation in the wild. In *The IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 1, 2, 6

[36] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. It's written all over your face: Full-face appearance-based gaze estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 2299–2308, 2017. 5, 6

[37] Xucong Zhang, Michael Xuelin Huang, Yusuke Sugano, and Andreas Bulling. Training person-specific gaze estimators from user interactions with multiple devices. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 2018. 1

[38] Xucong Zhang, Yusuke Sugano, and Andreas Bulling. Revisiting data normalization for appearance-based gaze estimation. In *Proceedings of the ACM Symposium on Eye Tracking Research & Applications*, 2018. 5

[39] Xucong Zhang, Yusuke Sugano, and Andreas Bulling. Evaluation of appearance-based methods and implications for gaze-based applications. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019. 1, 3

3D Prior is All You Need: Cross-Task Few-shot 2D Gaze Estimation

Supplementary Material

6. Summary of Evaluation Datasets

Our work conducts experiments on three datasets: MPIIGaze, which is also referred to as MPIIFaceGaze, EVE, and GazeCapture. We preprocess these datasets for evaluation. The dataset statistics are summarized in Table 6. Notably, GazeCapture includes over 1,000 subjects, making it impractical to evaluate all subjects. Therefore, we sort all subjects based on their identifiers, e.g., sub_00001, and select the 20 subjects in ascending order of their identifiers. We exclude subjects with fewer than 500 images to ensure a convincing evaluation.

Overall, under the experimental settings, our method was evaluated on 74 subjects across three different platforms, demonstrating its advantages and robustness.

Table 6. Dataset statistics on our experiment

	Devices	# Subjects	# Images per subject
EVE	Desktop computer	39	$\sim$ 1800
MPIIGaze	Laptop	15	1500
GazeCapture	Phone & tablet	20	$\sim$ 1200

Besides, we perform image normalization during data preprocessing. The original method requires camera intrinsic parameters for normalization, which conflicts with one of our motivations: enabling quick adaptation for non-expert users. In our method, this issue is resolved by using estimated camera intrinsic parameters. For the GazeCapture dataset, we apply estimated parameters for normalization. We do not provide detailed explanations of this in our manuscript, as it is not a central focus of our work and can be addressed effectively.

7. Implementation Details

Our work is primarily implemented using two libraries: PyTorch and PyTorch3D. Most of our modules are developed using PyTorch, while the Rodrigues transformation is implemented using PyTorch3D. The Rodrigues transformation ensures that $\mathbf{R} \in SO(3)$, facilitating the computation of the inverse matrix for coordinate transformations. We initialize the rotation matrix $\mathbf{R}$ as diag(-1, 1, -1) and the translation vector $\mathbf{t}$ as (0, 0, 0), where the value of $\mathbf{r}$ could be computed using Rodrigues formula. This represents a basic transformation between the camera coordinate system and the screen coordinate system, i.e., we assume that the origins of the two systems overlap, and the x-y planes of the two systems are parallel.

This is also why we claim that the initial screen pose happened to be same as the actual screen pose in the GazeCapture dataset. The dataset collects images using mobile devices, where the x-y planes of the embedded camera are typically parallel to the screen. More importantly, the authors of GazeCapture precisely measure the camera placement and screen dimensions to define a unified prediction space, setting the origin of the defined screen coordinate system at the camera position. However, this setup is atypical, as manually measuring the camera placement is equivalent to manually calibrate screen pose.

8. Implementation on the Real-World Device

We also evaluate our method in a real-world environment. Specifically, we implement our method at a desktop computer and invite a volunteer for testing. Our implementation is conducted on the machine equipped with an NVIDIA RTX 3090 GPU in a Python environment. We use a 1080p webcam to capture the face images and applied our method to estimate the user's 2D gaze on the screen. The experimental setup is illustrated in Figure 8, with the user positioned approximately 90 cm from the camera.

Figure 8. Setup for the implementation on the real-world device.

The process involves the following steps:

1. Calibration: The volunteer is required to look at four calibration points on the screen. For each point, we collect 40 face images for model adaptation while the volunteer focuses on each point for 1-2 seconds. We collect 40 images from each point to minimize the impact of noise data such as blink.
2. Data Pre-Processing: From these images, we detect human face landmarks and estimate the 3D head pose to compute the 3D face centers. Image normalization is then applied to obtain the processed face images.
3. Model Adaptation: Using our method, we adapt the 3D gaze estimation model for the real-world 2D gaze estimation.

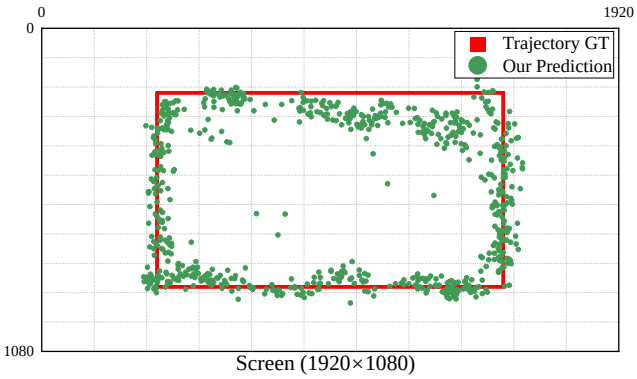

Figure 9. The visualization of our estimation in a real-world implementation. The red line represents the trajectory of a moving dot that we pursued, while the blue dots indicate our gaze predictions. This result demonstrates the effectiveness of our method. The jitter is due to various environmental and personal factors, such as eye blinking and unstable face detection. These issues can be easily addressed using post-processing methods.

4. Evaluation: The volunteer is instructed to continuously focus on a moving dot. Our method estimates the gaze trajectory from face images.

We visualize the results in Figure 9, where the red line represents the trajectory of the moving dot, and the green dots indicate the gaze estimation results. It is important to note that we do not pre-calibrate the camera intrinsic matrix or the screen pose. Besides, we do not apply post-processing methods, such as blink detection or filtering, to the estimation results. A video of the entire process is provided as supplementary material.