2025 IEEE International Symposium on Circuits and Systems (ISCAS)

FrameVoting: A Robust and Fast Method of Using Gaze Estimations to Identify Objects of Interest

Kao Den Chang
Harvard University
kaodenchang@fas.harvard.edu

He-Yen Hsieh
Harvard University
heyenhsieh@g.harvard.edu

H. T. Kung
Harvard University
kung@harvard.edu

Ziyun Li
Meta Reality Labs
liziyun@meta.com

Sai Qian Zhang
New York University
sai.zhang@nyu.edu

***Abstract*—We introduce FrameVoting, a voting-based method for real-time, gaze-driven object identification. It is a training-free method that incurs small computation and low processing latency, making the method ideal for wearable devices. In FrameVoting, the Point of Gaze (PoG) in each frame is used to define a potential region of interest. Regions across multiple frames are compared using the Sum of Absolute Differences (SAD) as a similarity measure. Each frame votes for the region from each of the other frames that is most similar to the region in the current frame, and only the region receiving the most votes is considered as the user's region of interest and sent to a classifier for inference. FrameVoting thus eliminates the need for frame-by-frame bounding box retrieval and object detection required by traditional methods, thereby reducing computation overhead and latency. The method is robust, as it eliminates the need for threshold-tuning to determine whether gaze estimations are focused on a specific object. Further, the method is efficient and fast, as inference is only performed on the most-voted region, and the SAD computation is highly parallelizable. Our experiments on the AEA Dataset demonstrate that FrameVoting reduces the frequency of inferences by 95.6% compared to frame-by-frame object detection, while still accurately identifying the user's objects of interest in real-time at 30 fps on a Raspberry Pi 5.**

I. Introduction

Modern wearable devices, such as AR/VR headsets, increasingly rely on high-frequency gaze estimations in user interaction. However, performing object detection in every frame on the region where the user looks introduces significant computation overhead and leads to high latency [1]. Additionally, conventional object detection methods require substantial processing power for object classification, posing challenges for devices with limited computational and battery resources.

To address these limitations, we propose FrameVoting, a voting-based method that identifies objects of interest based on the user's gaze. Instead of performing object detection on every frame, our approach classifies objects only when the gaze consistently focuses on a specific region, indicating that the user is actively observing an object. By reducing the frequency of inferences, FrameVoting significantly lowers computation costs and latency, while also extending battery life. Thus, the method enables real-time, on-device inference without offloading computation from wearable devices. Furthermore, our approach is compatible with existing gaze estimation models [2] to determine the Point of Gaze (PoG) for gaze-driven object detection.

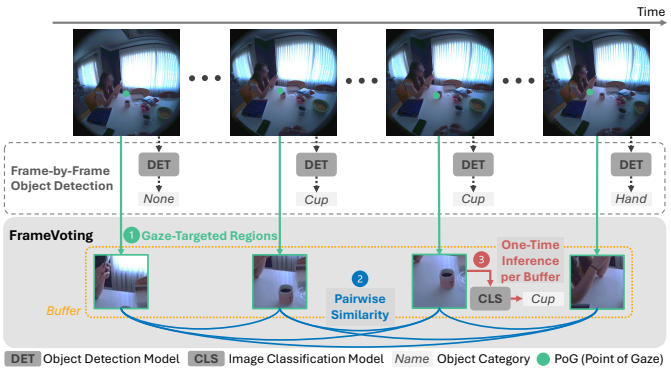

Fig. 1: **FrameVoting Performs Inference Based on User Gaze Estimations:** By utilizing gaze-targeted regions (green label 1) and computing pairwise region similarities (blue label 2), FrameVoting performs only one inference (red label 3) on the region that is most similar to others in the buffer, using a voting mechanism. In this demonstration, the third region receives the most votes (as shown in the third row), thereby eliminating the need for frame-by-frame object detection (depicted in the second row). This approach significantly reduces overall computational cost and latency.

To motivate the technical approach of this paper, we first present some background information on human eye behavior. Human gaze tends to quickly focus and stabilize on objects of interest, even after brief distractions [3]. FrameVoting emulates this behavior by analyzing gaze-targeted *regions* over time, rather than performing costly frame-by-frame inferences, which are impractical for wearable devices. During distractions, gaze-targeted regions generally exhibit low similarity across frames and can thus be disregarded. Accordingly, FrameVoting performs inference computations only on regions where the gaze is consistently directed.

Specifically, FrameVoting examines gaze-targeted regions for efficient object detection. In each frame, a potential region of interest is defined based on the PoG, and these regions are stored in a buffer for comparisons. Regions across multiple frames are compared using the Sum of Absolute Differences (SAD) as a similarity measure, with each region voting for the one most similar to itself among regions in the buffer.

The region receiving the most votes is then selected for classification and considered as the user's object of interest, as shown in Figure 1.

Furthermore, FrameVoting eliminates the need for threshold tuning [4], [5] in determining whether the user is focusing on a specific object. These thresholds are difficult to calibrate without prior knowledge of individual eye movement patterns, which can vary significantly among users [2], [6]. In contrast, FrameVoting uses a voting mechanism to eliminate the need for such ad-hoc threshold tuning, allowing the system to adapt seamlessly to different gaze behaviors without requiring calibration. Additionally, parallel computation of SAD enhances processing speed, resulting in higher frame rates and lower latency, both crucial for reliable performance in real-time wearable applications.

FrameVoting introduces the following contributions:

1) FrameVoting eliminates the need for continuous frame-by-frame object detection by only comparing regions across frames and performing a single object classification. This approach significantly reduces computational cost, inference time, and latency, making it ideal for low-power applications.
2) Being training-free, FrameVoting is compatible with existing gaze estimation models [2]. This ensures seamless integration with current tools and systems.
3) Our experiments using the AEA dataset demonstrates that FrameVoting reduces the frequency of inferences by 95.6% compared to frame-by-frame object detection.

II. Related Work

Recent advancements in wearable devices, such as AR/VR headsets, have increasingly relied on high-frequency gaze estimations, reaching up to 90 fps [7]. Despite these advancements, integrating real-time object detection on such devices remains challenging due to the high computational demands of current methods [8]. These methods often lead to increased latency and rapid battery depletion, rendering them impractical for devices with limited resources.

A common approach to addressing these challenges is to offload the computational burden to the edge or cloud [1]. While offloading alleviates on-device processing, it introduces substantial bandwidth overhead, often exceeding 60 Mbps, thereby diminishing efficiency gains. Furthermore, communication delays adversely affect the real-time performance required for gaze-driven object dectection on wearable devices.

Fang et al. [5] proposed tuning-based methods that depend on additional training and tuning phases, achieving a performance of only 1 fps, which is insufficient for real-time applications. This reliance on model-specific tuning increases complexity and reduces adaptability, making these methods unsuitable for low-power, real-time use.

In contrast, FrameVoting introduces a training-free, voting-based approach that eliminates the need for threshold tuning, thus lowering computational costs and latency. By leveraging parallelizable SAD computation, it minimizes overhead and achieves real-time gaze-driven object identification at 30 fps.

III. Efficient Gaze-Driven Object Identification

This section presents FrameVoting, a method designed to accelerate gaze-driven object identification by exploiting gaze stability and employing efficient region selection. FrameVoting is compatible with existing gaze estimation models [2] for determining the Point of Gaze (PoG). Object detection involves identifying gaze-targeted regions and subsequently classifying them. Section III-A discusses the collection of gaze-targeted regions across consecutive frames using a buffer. Section III-B elaborates on the voting mechanism, which employs pairwise similarity using SAD to classify consistently viewed regions. Section III-C focuses on the parallel processing of SAD, facilitating high frame rates and low latency.

A. Gaze-Targeted Region Accumulation in a Buffer

Detecting objects of interest based on gaze often relies on thresholds such as eye stability criteria, fixation duration, and saccade velocity. Calibrating these thresholds is challenging due to variations in individual eye movement patterns [2], [4], [6]. Such variations complicate the detection process, as thresholds may not generalize well across users. Furthermore, continuously detecting object locations across frames can introduce bottlenecks, thereby increasing system latency and affecting real-time performance.

To address these challenges, FrameVoting leverages gaze stability to improve object detection. Instead of performing detection on every frame, FrameVoting collects gaze-targeted regions from consecutive frames using a buffer. These regions are obtained by cropping a small area, one-fifth of both the frame height and width, centered on the PoG in each frame, as illustrated in Figure 2. The buffer takes advantage of the eye's natural tendency to refocus on an object over time [3], with the accumulated regions reflecting the user's intent. The buffer size (N) is determined by the frame rate of gaze estimations (F) and the interval at which the human eye refocuses on the object of interest, denoted as S seconds, ensuring that a majority of gaze-targeted regions in the buffer capture the intended object. Specifically, the buffer size is defined as:

$$N \gtrapprox 2 \times S \times F \tag{1}$$

This equation enables FrameVoting to effectively capture the user's object of interest. By spanning more than twice the time required for the human eye to refocus, the buffer primarily contains frames corresponding to the object of interest, thereby making the system robust to brief distractions. Given a typical refocusing time of 0.2 seconds (S) and a frame rate of 30 fps (F), the buffer size (N) is set to 12 gaze-targeted regions, as illustrated in Figure 4.

B. Voting-Based Region Classification

Pairwise Similarity: After accumulating the gaze-targeted regions in the buffer, FrameVoting computes the pairwise similarity between these regions using SAD, formulated as:

$$SAD_{ij} = \sum |\mathbf{I}_i^R - \mathbf{I}_j^R| \tag{2}$$

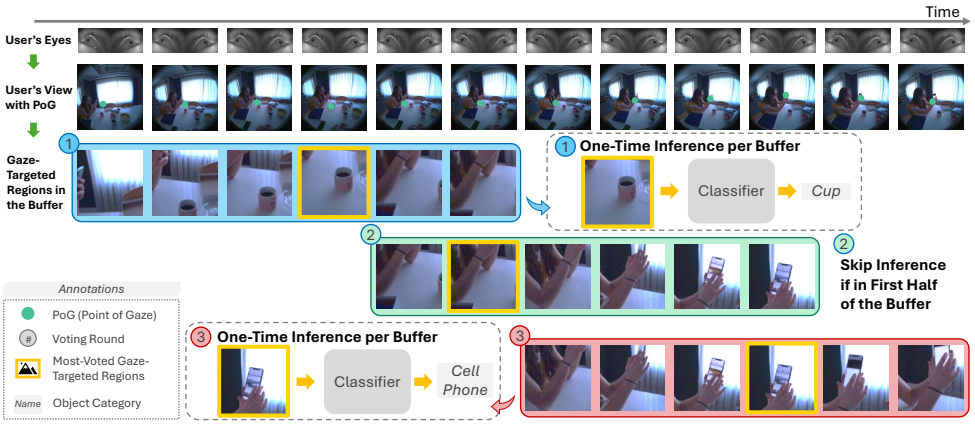

Fig. 2: **FrameVoting Illustration of Performing One-Time Inference Per Buffer:** The top row displays the user's eye movement, followed by the PoG projected onto the user's view in the second row. FrameVoting uses a buffer to gather gaze-targeted regions. Within the buffer, similarities between regions are computed, and only the region with the most votes (highlighted by a yellow border) is forwarded to the classifier for region classification, as depicted in the third row. The buffer is then shifted to collect new cropped regions, starting just after the frame with the most votes. As indicated in the fourth row, if the region with the most votes is located in the first half of the buffer, classification is skipped. The process continues with shifting the buffer, and classification is performed on the most-voted region in the final row.

where $\mathbf{I}_i^R$ and $\mathbf{I}_j^R$ represent the i-th and j-th regions in the buffer. Each region is compared to every other region, with the one exhibiting the lowest SAD value deemed the most similar. In this approach, each region votes for the region most similar to itself within the buffer. The region that receives the most votes, due to its similarity across comparisons, is selected as the target region, as illustrated in Figure 3.

Inference Skipping: A region with the most votes is selected for one-time classification inference if it is located in the second half of the buffer and exhibits a significant statistical difference from the previously inferred region, determined through voting. Conversely, inference is skipped to prevent redundant computations. This process ensures that FrameVoting performs region classification only when there is a consistent user focus on a region that differs significantly from the previously inferred region, thereby reducing unnecessary computations and improving overall efficiency.

FrameVoting Demonstration: The last three rows in Figure 2 illustrate the FrameVoting process. For simplicity, this example uses a buffer size of 6 gaze-targeted regions. FrameVoting computes the pairwise similarity of the accumulated gaze-targeted regions within the buffer. The region with the most votes (highlighted by a yellow border) is selected and passed to the classifier for region classification, provided it is not skipped due to the criteria mentioned in the previous section. The buffer is then shifted forward to collect new gaze-targeted regions, beginning immediately after the time frame with the most votes. This process continues, ensuring efficient region classification, with classification performed on the most-voted region only when necessary.

C. Parallelization of SAD Computation

SAD is highly parallelizable [9], making it ideal for real-time applications. The absence of data dependencies in pair-

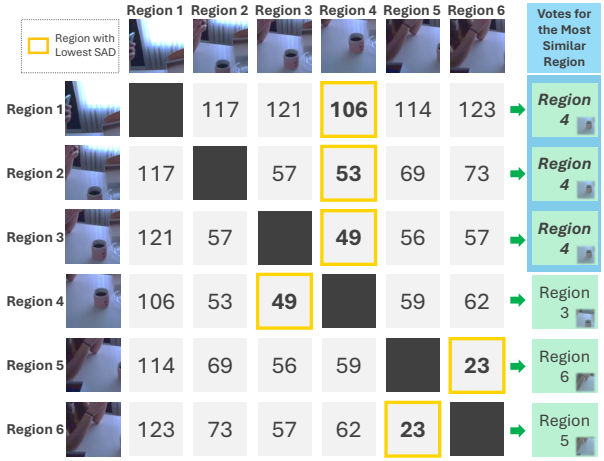

Fig. 3: **Voting Mechanism Based on SAD Computation:** The table displays candidate regions in the top row and leftmost column, with pairwise SAD scores listed within, expressed in units of 10,000. For each region, the lowest SAD score in its row indicates the region to which it is most similar. The rightmost column summarizes the voting results, showing that Region 4 receives the most votes, being selected three times.

wise SAD calculations allows for efficient parallel processing. Further acceleration is achieved by fusing subtraction, absolute, and summation operations into a single operation using CPU intrinsics [10]. This optimization enhances FrameVoting's efficiency, enabling high frame rates and low latency.

IV. Experiments

A. Experimental Setup

We evaluate FrameVoting using the Aria Everyday Activities (AEA) dataset [7], which provides diverse gaze scenarios

Method	# of Inferences	Reduction of Inferences	FPS
Baseline (Frame-by-Frame)	1,293	-	1
FrameVoting	**57**	**95.6%**	**30**

TABLE I: **Comparison of Conventional Frame-by-Frame Object Detection and FrameVoting:** The table presents the number of inferences, reduction in inferences, and frame rate (fps), demonstrating how FrameVoting significantly reduces the number of inferences and improves the frame rate.

through eye images captured with Project Aria glasses during various daily activities. YOLOv7-tiny [11] is employed as the object detector throughout the experiments. All tests are conducted on a Raspberry Pi 5, without any additional acceleration modules, with frames input at a rate of 30 frames per second.

B. Gaze-Driven Object Detection Comparisons

Table I compares the baseline frame-by-frame object detection method with FrameVoting on the AEA dataset. Both methods utilize the same object detector, YOLOv7-tiny, to ensure consistency. In the baseline approach, object detection is performed on every frame using the full frame dimensions to retrieve bounding boxes and classification results, resulting in high computational costs. In contrast, FrameVoting focuses solely on gaze-targeted regions, each sized at one-fifth of the original frame's height and width, using the same detector solely for region classification. This approach eliminates the need for bounding box retrieval and frame-by-frame inference, significantly reducing computation time. By avoiding frame-by-frame inference, FrameVoting achieves a 95.6% reduction in the number of inferences (from 1,293 to 57). Consequently, this reduction leads to a substantially higher frame rate, enabling FrameVoting to maintain real-time performance at 30 fps, compared to the baseline method's 1 fps.

C. Ablation Study

Buffer Size and Inference Precision: Figure 4 examines the effect of buffer size (N in Equation 1) on the effectiveness of FrameVoting in identifying the user's objects of interest in videos. This analysis highlights how accurately the selected region matches the object of interest. We introduce the metric Inference Precision, defined as:

$$\text{Inference Precision} = \frac{\text{Effective Inferences}}{\text{Total Inferences}}$$

where Effective Inferences occur only when FrameVoting correctly identifies an object of interest that the user is focusing on, counted only once per continuous focus on the same object. Our analysis shows that a buffer size of 12 achieves the best results in terms of precision, under the conditions specified in Equation 1.

FrameVoting Configurations: Table II evaluates the effectiveness of various FrameVoting configurations based on the number of inferences. Row 1 presents the Baseline (Frame-by-Frame) approach, where object detection is performed on every frame, resulting in 1,293 inferences, making it the least efficient setting. Row 2 represents the base FrameVoting configuration, where gaze-targeted regions are accumulated in a buffer, and the most-voted region from each buffer is selected for classification. This setup requires 305 inferences, achieving a 73.6% reduction in inferences compared to the baseline. To further improve efficiency, redundant region classifications can be avoided by skipping the most-voted region in the subsequent buffer when applicable. Row 3 introduces Similarity-Based Skipping, where inferences are skipped if the most-voted region is similar to the previously selected region. This approach reduces the number of inferences to 248, resulting in an 80.8% reduction. Row 4 applies Buffer-Based Skipping, where inferences are skipped for regions located in the first half of the buffer. This method reduces the number of inferences to 60, achieving a 95.4% reduction. Finally, Row 5 combines both Similarity-Based Skipping and Buffer-Based Skipping, further reducing the number of inferences to 57. This achieves a 95.6% reduction compared to the baseline.

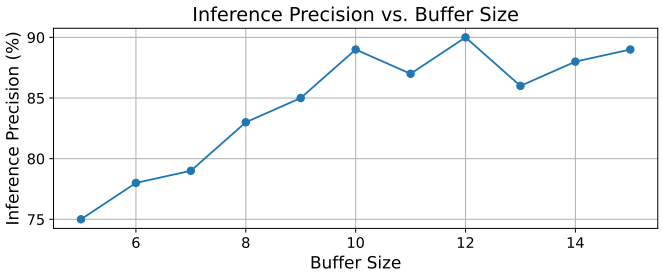

Fig. 4: **Ablation Study on Buffer Size v. Inference Precision**

Row	Method	Skip Most-Voted Regions		# of Inferences (Reduc.)
		Similarity-Based	Buffer-Based	
1	Baseline (Frame-by-Frame)	-	-	1,293 (-)
2	**FrameVoting**	-	-	305 (↓ 76.4%)
3		✓	-	248 (↓ 80.8%)
4		-	✓	60 (↓ 95.4%)
5		✓	✓	**57 (↓ 95.6%)**

TABLE II: **Ablation Study Comparing Baseline and FrameVoting Configurations:** The table highlights the applied skipping methods, the resulting number of inferences, and the respective reduction percentages.

V. Conclusion

In this paper, we introduce FrameVoting, an efficient method for real-time gaze-driven object identification. By exploring the stability that the Point of Gaze (PoG) may present, FrameVoting significantly reduces computational overheads. The method uses a voting mechanism based on the Sum of Absolute Differences (SAD) to select only the gaze-targeted region with the highest similarity across frames to perform inference. Our evaluation on the AEA Dataset demonstrated that FrameVoting reduces the frequency of inferences by 95.6% compared to conventional methods. The parallel computation of SAD further speeds up the processing to achieve real-time performance at 30 fps on a Raspberry Pi 5, making FrameVoting ideal for resource-limited wearable devices.

Acknowledgment

This research was supported in part by the Air Force Research Laboratory under award number FA8750-22-1-0500, and in part by Meta Platforms Technologies under award number A51540.

References

[1] L. Liu, H. Li, and M. Gruteser, "Edge assisted real-time object detection for mobile augmented reality," in *The 25th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '19, New York, NY, USA: Association for Computing Machinery, Aug. 5, 2019, pp. 1–16.

[2] R. Kothari, Z. Yang, C. Kanan, R. Bailey, J. B. Pelz, and G. J. Diaz, "Gaze-in-wild: A dataset for studying eye and head coordination in everyday activities," *Scientific Reports*, vol. 10, no. 1, p. 2539, Feb. 13, 2020.

[3] T. Tammi, J. Pekkanen, S. Tuhkanen, L. Oksama, and O. Lappi, "Tracking an occluded visual target with sequences of saccades," *Journal of Vision*, vol. 22, no. 1, Jan. 2022.

[4] L. Chukoskie, S. Guo, E. Ho, *et al.*, "Quantifying gaze behavior during real-world interactions using automated object, face, and fixation detection," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 10, no. 4, pp. 1143–1152, Dec. 2018.

[5] W. Fang and K. Zhang, "Real-time object detection of retail products for eye tracking," in *2020 8th International Conference on Orange Technology (ICOT)*, Dec. 2020, pp. 1–4.

[6] H. Nurlatifa, R. Hartanto, A. Ataka, and S. Wibirama, "Optimum object selection methods for spontaneous gaze-based interaction with linear and circular trajectories," *Results in Engineering*, vol. 21, p. 101 769, Mar. 1, 2024.

[7] Z. Lv, N. Charron, P. Moulon, *et al.*, *Aria everyday activities dataset*, 2024.

[8] "Object tracking for visionOS is here - but how does it work?" LEARN XR BLOG, Available: https://blog.learnxr.io/xr-development/apple-object-tracking-for-visionos-is-here.

[9] A. Medhat, A. Shalaby, M. S. Sayed, M. Elsabrouty, and F. Mehdipour, "A highly parallel SAD architecture for motion estimation in HEVC encoder," in *2014 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, Nov. 2014, pp. 280–283.

[10] S. Manilov, B. Franke, A. Magrath, and C. Andrieu, "Free rider: A source-level transformation tool for retargeting platform-specific intrinsic functions," *ACM Trans. Embed. Comput. Syst.*, vol. 16, no. 2, 38:1–38:24, Dec. 12, 2016.

[11] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, ISSN: 2575-7075, Jun. 2023, pp. 7464–7475.