Foveated Instance Segmentation

Hongyi Zeng[†] Wenxuan Liu[†] Tianhua Xia
New York University

Jinhui Chen Ziyun Li
Meta Reality Labs

Sai Qian Zhang
New York University

Abstract

Instance segmentation is essential for augmented reality and virtual reality (AR/VR) as it enables precise object recognition and interaction, enhancing the integration of virtual and real-world elements for an immersive experience. However, the high computational overhead of segmentation limits its application on resource-constrained AR/VR devices, causing large processing latency and degrading user experience. In contrast to conventional scenarios, AR/VR users typically focus on only a few regions within their field of view before shifting perspective, allowing segmentation to be concentrated on gaze-specific areas. This insight drives the need for efficient segmentation methods that prioritize processing instance of interest, reducing computational load and enhancing real-time performance.

In this paper, we present a foveated instance segmentation (FovealSeg) framework that leverages real-time user gaze data to perform instance segmentation exclusively on instance of interest, resulting in substantial computational savings. Evaluation results show that FSNet achieves an IoU of 0.56 on ADE20K and 0.54 on LVIS, notably outperforming the baseline. The code is available at https://github.com/SAI-Lab-NYU/Foveated-Instance-Segmentation

1. Introduction

Semantic segmentation [5, 23, 29, 49], a fundamental task in computer vision, involves partitioning an image into meaningful regions to facilitate the analysis and interpretation of its visual content. Instance segmentation [6, 15, 46, 51] takes this further by identifying and delineating each individual object instance within an image, which plays a critical role in augmented reality (AR) as it enables precise object recognition and separation in real-world scenes, allowing for more accurate interaction, manipulation, and integration of virtual elements with physical objects for an immersive and context-aware user experience .

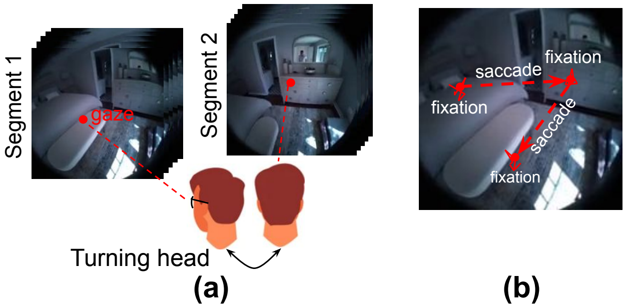


Figure 1. (a) An example on gaze location for the AR user. (b) Trace of eye gaze within a segment.

Despite its importance, the segmentation task poses substantial computational challenges, particularly on resource-limited AR/VR devices, largely due to the high resolution of input images captured by these devices. For instance, the Meta Ray-Ban glasses feature a 12-megapixel camera capable of recording 1440P video [34], which results in significant computational overhead during instance segmentation. This high data volume results in considerable computational latency, which can severely limits performance and responsiveness, ultimately degrading the overall user experience by impeding real-time interaction and fluidity.

In contrast to conventional use cases, AR/VR device users have a unique behavior: they tend to focus on only some small areas within a view before shifting to another view. For instance, as shown in Figure 1, a user wearing AR glasses stands in a bedroom. In the left part of Figure 1 (a), the user looks at the bed for a few seconds before turning head to look at the wardrobe, as depicted in the right part of Figure 1 (a). In this scenario, the sequential video frames can be divided into two segments based on head movement. Within the first segment, where the frames are highly similar, the gaze is primarily focused on the bed, allowing instance segmentation to be performed only on the bed. Similarly, in the second segment, segmentation can be limited to the wardrobe. This insight offers an inherently efficient solution for instance segmentation in AR/VR environments by prioritizing the processing of instances of in-

[†] Equal Contribution
Experiments were conducted at New York University (NYU).

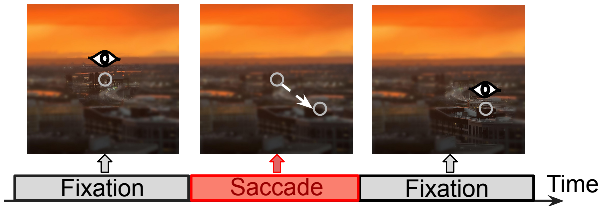


Figure 2. An example on fixation and saccade of human eye.

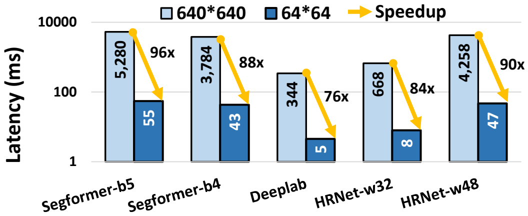


Figure 3. Processing latency of segmentation task on edge GPUs.

terest (IOI) as determined by the user's gaze. By focusing computational resources on these targeted areas, it is possible to significantly reduce processing workload and computational costs, enhancing the real-time performance of AR/VR applications and improving the overall user experience. This aligns naturally with the paradigm of *foveated rendering* [36], which enhances graphical performance by rendering images at full resolution only in the area where the user's gaze is directed, while reducing detail in the peripheral vision to conserve computational resources.

In this paper, we propose a novel approach to instance segmentation, termed *foveated instance segmentation*, by adopting a foveated processing strategy, where segmentation is applied solely at the instance where human gaze locates, eliminating the need to process the entire image. While this approach holds significant potential for efficiency, it also presents several challenges. The first challenge is designing a deep neural network (DNN) framework capable of processing only the IOI associated with the gaze location. The second challenge involves leveraging the temporal dynamics of human gaze to further reduce redundant computations and enhance processing efficiency. Our contribution can be summarized as follow:

- We propose a novel and crucial insight into instance segmentation that leverages human eye behavior to reduce computational costs in AR/VR environments.
- We introduce *FSNet*, a plug-and-play instance segmentation neural network that takes a high-resolution input image and gaze location, efficiently performing instance segmentation solely on the instance of interest (IOI). FSNet can integrate with any existing segmentation network, significantly enhancing its generalizability.
- Building on FSNet, we further introduce *FovealSeg*, an efficient foveated instance segmentation framework designed for real-time AR/VR processing. FovealSeg leverages temporal similarity between consecutive frames and human gaze patterns to optimize segmentation, enhancing performance for dynamic AR/VR environments.

2. Background and Related Work

2.1. Human Eye Behavior

The human eye functions in three primary modes of movement, each with distinct roles: *fixation*, when the eye is still and focused on a single point; *saccadic movements*, rapid, jerky movements that shift the gaze from one target to another; and *smooth pursuit*, when the eye smoothly follows a moving object. Among these, smooth pursuit is less common, while fixation and saccadic movements dominate most of our visual activity, as shown in Figure 2. During the fixation, human gaze remains focused around a single point, with visual acuity mostly concentrates at the region around the gaze location, and drops greatly outside the region, leading to decreased perception in peripheral vision [28].

Additionally, humans make one to three saccadic eye movements per second [12, 24, 25], with each saccade duration typically ranges from 20-200ms and speed reaching speeds over 200° per second [39]. Human saccade are essential for scanning the environment efficiently. During a saccade, visual information becomes momentarily blurred due to the high speed of the eye movement, a phenomenon known as saccadic blur [7, 18, 33]. Once the eye reaches its new target, visual clarity is restored, and the brain integrates information from both the fixation and motion periods to create a stable perception of the environment. Previous studies have shown that perceptual saccadic suppression reaches its peak at the maximum velocity of saccadic movement, leading to a substantial reduction in detectability by at least 75% [18]. This allows visual operations to be temporarily halted during the saccade stage without affecting the user's visual experience, as demonstrated in [22, 30].

Figure 1 (b) shows an example on trace of the human gaze within a single frame segment. Initially, the gaze is fixed on the bed, followed by a saccade where it rapidly shifts to the wardrobe. During the fixation stage, minor movements may occur due to natural eye muscle activity. In this context, only two instances need to be detected (i.e., the bed and the wardrobe), and processing during the saccade phase is unnecessary, as human vision is less sensitive and the gaze moves rapidly during saccades. Previous studies [3, 4] have shown that saccades can be reliably detected by analyzing gaze location changes over a unit time period. When this change surpasses a certain high threshold, a saccade is identified. A similar criterion can be applied to detect the end of a saccade.

2.2. Instance Segmentation Cost in AR/VR device

Since modern AR/VR devices do not allow users to modify the internal algorithm, we use the GPU simulation tool GPGPU-Sim [21] to simulate hardware performance. We configure GPGPU-Sim to model the Jetson Orin NX [1], a widely used edge GPU in AR/VR environments [13, 16, 35, 52, 53]. We simulate the hardware performance by measuring the processing latency of four popular deep neural network (DNN) architectures for segmentation: Segformer [49], Deeplab [8], and HRNet [45]. As shown in Figure 3, applying semantic segmentation to an input size of 640 × 640 results in processing latencies exceeding 300 ms, with some architectures, such as Segformer, experiencing delays of over a second. This substantial processing time leads to serious latency and a noticeable drop in user experience, as achieving 10-20 frames per second (FPS), namely a processing latency of 50-100ms, is generally required for a satisfactory visual experience [2]. In contrast, reducing the input size to 64 × 64 greatly lowers processing latency, meeting the user requirement for speed, but on the other hand will degrade the accuracy performance.

In addition, prior studies [26, 37, 47, 48, 50, 51] on video instance segmentation process consecutive frames jointly (e.g., VisTR [50] and SeqFormer [48] process 5 consecutive frames together). This method leverages temporal correlations across frames to enhance performance; However, this approach introduces significant latency because processing does not begin until all frames are available, leading to a substantial delay that hinders real-time responsiveness. For example, a Meta Ray-Ban device running at 30 FPS [34] would experience a delay of $5 \times \frac{1}{30} = 167$ms, which significantly exceeds the acceptable processing latency range of 50-100 ms [2].

2.3. Gaze Behavior in AR/VR Environment

In this section, we present a detailed study of gaze behavior using real-world data from the Aria Everyday Activities Dataset [31], which captures sequences of images aligned with the user's field of view and tracks gaze movement within each frame, as illustrated in Figure 4 (a).

To quantify head movement, we compute the image difference by calculating the Euclidean distance between corresponding pixels of consecutive frames. If a user maintains a steady gaze in one direction for a period of time, the resulting pixel differences between frames will be minimal. When the user shifts their head, the pixel difference increases, indicating movement. The left part of Figure 4 (c) represents the pixel difference over a sample time interval. To identify frames with minimal changes, we set a threshold for the pixel difference. If the pixel difference falls below this threshold, the frames will be highly similar and imperceptible to the human eye, allowing them to be grouped into a single **segment**, as shown by the gray shading in the left part of Figure 4 (b) and (c).

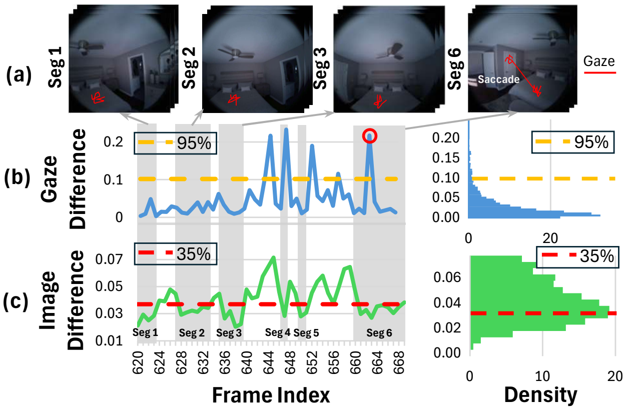


Figure 4. (a) Images and corresponding gaze locations from the Aria Everyday dataset [31], collected from real users wearing a VR headset. (b) Left: Changes on gaze locations over the frames. Right: Histogram of gaze differences with the yellow line marking the 95% threshold. (c) Left: Normalized pixelwise differences across frames, with gray blocks indicating frames within the same segments. A 0.037 threshold is used to group similar frames. Right: The histogram of image differences. 35% of pairs of consecutive frames with a difference less than 0.037.

Within each segment, segmentation results can be reused if the gaze remains relatively stable. To demonstrate this, we analyze gaze location difference within a segment, as shown in the left part of Figure 4 (b). The results indicate that a threshold of 0.1 can be used to group gaze locations within the fixation phase; values above this threshold suggest the occurrence of a saccade, as shown in Segment 6 of Figure 4 (a). Additionally, 95% of frames within each segment have a gaze difference below 0.1. This study shows significant potential to improve the computational efficiency of instance segmentation tasks by **focusing processing solely on the IOI and reuse the results within a segment**.

2.4. Gaze Tracking System in AR/VR Devices

Gaze tracking is pivotal in AR/VR systems, as it provides a seamless and natural interface by accurately identifying where users are focusing their attention within immersive environments. Most commercial AR/VR devices are equipped with integrated eye-tracking systems. For example, popular headsets such as the Meta Quest Pro [11] and HTC Vive Pro Eye [17] come with built-in eye trackers that monitor users' gaze in real-time, achieving latencies as low as 5-10 milliseconds and sampling rates up to 120 Hz [17, 42]. These eye-tracking modules use infrared or image sensors to capture eye movements with high accuracy and speed, enabling the system to determine where the user is looking at any given moment.

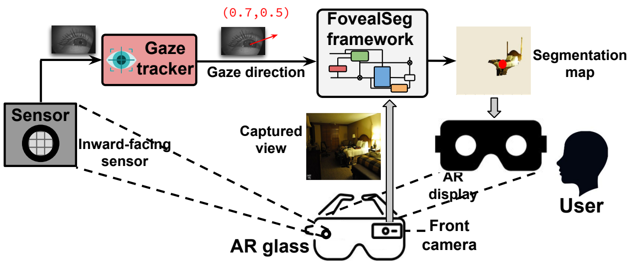


Figure 5. An overview of FovealSeg framework.

2.5. Image Segmentation via Input Downsampling

Previous research has focused on creating efficient input downsampling methods for DNNs. By making downsampling differentiable, these methods enable the training process to adjust sampling resolution selectively, enhancing performance while reducing input dimensions and improving efficiency. In [38], the authors present a saliency-based distortion layer for convolutional neural networks that enhances spatial sampling of input data for image classification tasks. Subsequent works, such as [19, 32, 43], apply similar concepts by learning a saliency score for each pixel to guide the downsampling process, resulting in improved performance for semantic segmentation tasks. However, while the zoom process is learnable, the unzoom process, which projects the label maps back to the original dimensions, is often performed using an analytical solution, leading to suboptimal results. To address this, LZU [44] proposes an efficient solution to learn both the zoom and unzoom processes, streamlining the entire downsampling and upsampling workflow. In contrast, FSNet utilizes user gaze input along with the captured image to perform instance segmentation only at IOI. Building on FSNet, we propose FovealSeg framework to perform instance segmentation across multiple frames with high efficiency. This approach leverages temporal correlation and human gaze behavior to optimize processing, reducing redundant computations over consecutive frames.

3. Methodology

Figure 5 illustrates the computational flow of the FovealSeg framework. During operation, the inward-facing sensor of the AR/VR device continuously captures images of the user's eye and sends them to the gaze tracker, which estimates the gaze direction with high accuracy in approximately 5-10 milliseconds [17, 42]. The estimated gaze direction is then passed to FovealSeg, along with the captured high-resolution image from the front camera as an additional input. FovealSeg generates a segmentation map focused solely on the IOI and reuse across frames with similar gaze locations.

In this section, we start by outlining the preliminary of image sampling in Section 3.1, then provide a detailed description of the FSNet design in Section 3.2, and describe the FovealSeg framework in Section 3.3.

3.1. Preliminary

Image downsampling can be viewed as an operation that transforms the original image $F \in \mathbb{R}^{H\times W\times C}$ into a new image $\hat{F} \in \mathbb{R}^{h\times w\times C}$, where $h \leq H$ and $w \leq W$. This downsampling process is achieved using two mapping functions, $g^h(.)$ and $g^w(.)$, which take the 2D coordinate (i,j) of the downsampled image $\hat{F}$ and produce the corresponding coordinate $(g^h(i,j), g^w(i,j))$ in the input image F. Thus, each pixel in $\hat{F}$ can be defined as:

$$\hat{F}[i,j] := F[g^h(i,j), g^w(i,j)] \tag{1}$$

where $F[i,j]$ denotes the pixel at coordinate (i,j) within F. Moreover, to eliminate the impact of the image dimension, instead of generating the actual coordinates, the mapping functions can process the normalized 2D coordinates. For example, let $G = \{G^h, G^w\}$ denote the set of mapping function, where both G^h and G^w takes a normalized 2D coordinate of $\hat{F}$, and produce normalized height and weight of X, respectively. Equation 1 can be rewritten as:

$$\hat{F}[i,j] := F\left[\left\lceil G^h\left(\frac{i}{h},\frac{j}{w}\right)H\right\rfloor, \left\lceil G^w\left(\frac{i}{h},\frac{j}{w}\right)W\right\rfloor\right] \tag{2}$$

where $\lceil\cdot\rfloor$ is the rounding function, $i \leq h$ and $j \leq w$. For instance, for uniform dowmsampling function, G^h, G^w are defined as follows:

$$\hat{F}[i,j] := F\left[\left\lceil \frac{i}{h}\times H\right\rfloor, \left\lceil \frac{j}{w}\times W\right\rfloor\right] \tag{3}$$

In saliency-guided downsampling [19, 44], the mapping operations G becomes learnable by incorporating a saliency-based sampling layer. In this layer, the saliency score, representing the relative sampling density at each normalized pixel location (i,j), is defined by $D_\theta(i,j)$. Here, $D_\theta(.)$ is a DNN that takes the original input image F and the normalized coordinates (i,j) to generate a score map with dimensions $\mathbb{R}^{h\times w\times 1}$. The parameter set θ represents the weights of the DNN. Given this, the mapping function can be determined as:

$$G^h(i,j) = \frac{\sum_{i',j'} D_\theta(i',j')k_\sigma((\frac{i}{h},\frac{j}{w}),(\frac{i'}{H},\frac{j'}{W}))i'}{\sum_{i',j'} D_\theta(i',j')k_\sigma((\frac{i}{h},\frac{j}{w}),(\frac{i'}{H},\frac{j'}{W}))} \tag{4}$$

$$G^w(i,j) = \frac{\sum_{i',j'} D_\theta(i',j')k_\sigma((\frac{i}{h},\frac{j}{w}),(\frac{i'}{H},\frac{j'}{W}))j'}{\sum_{i',j'} D_\theta(i',j')k_\sigma((\frac{i}{h},\frac{j}{w}),(\frac{i'}{H},\frac{j'}{W}))} \tag{5}$$

where $k_\sigma(x,x^{'})$ is a Gaussian kernel with a standard deviation of σ. To project $\hat{F}$ back to its original size, a reverse sampler G^{-1} restores the downsampled image $\hat{F}$ to

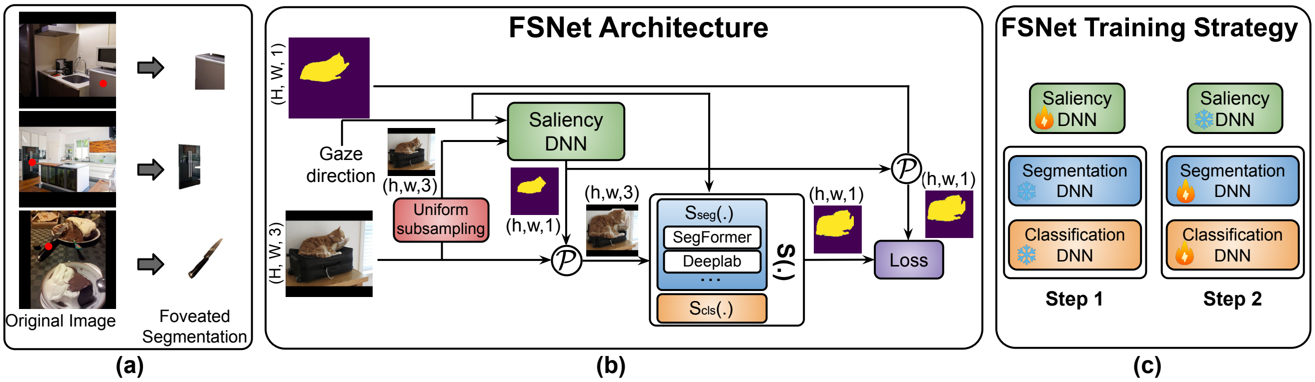


Figure 6. (a) An illustration of foveated instance segmentation, where the red point indicates the gaze focus region. In this approach, only the object within the gaze area is segmented, while all other regions are considered background. (b) Overview of the FSNet architecture during the training phase. $\mathcal{P}$ denotes the saliency-guided downsampling. (c) Alterative training strategy for FSNet.

the original space, using an interpolation function to compute the missing pixel values.

3.2. FSNet Training Methodology

Foveated instance segmentation poses two main challenges: (1) As illustrated in Figure 6 (a), foveated segmentation focuses on generating an instance segmentation mask solely for the IOI. Consequently, it requires creating a binary mask to identify the IOI region along with its associated class label. This differs from conventional segmentation approaches, which are designed for multi-class outputs. (2) Existing methods have difficulty utilizing gaze location as prior information to guide segmentation, complicating the task of distinguishing between foreground and background across various classes. Additionally, commonly used segmentation loss functions, such as standard joint loss and focal loss [19, 27], often fall short in fine-tuning when the target instance is particularly small. To address these issues, we introduce FSNet, which incorporates (1) a gaze-aware model architecture and (2) an optimized training and fine-tuning strategy.

Gaze-aware Model Architecture The FSNet architecture, as depicted in Figure 6 (b), begins with an input image $F \in \mathbb{R}^{H \times W \times 3}$. Incorporating a specified gaze location (u, v), we construct a *gaze map* N which is then concatenated to the input along the channelwise dimension. The map represents the normalized inverse distance to the gaze point, with higher values (closer to 1) signifying closeness to the gaze location, each element $N[i, j]$ of N is defined as follows:

$$N[i, j] = 1 - \frac{\sqrt{(i-u)^2 + (j-v)^2}}{d_{\text{max}}}$$

Here, $0 \leq i \leq H$ and $0 \leq j \leq W$. The term $d_{\text{max}} = \sqrt{H^2 + W^2}$ represents the maximum possible distance between any two pixels in F, serving as a normalization factor. We concatenate the gaze map with the input image and downsample the result as input to the saliency DNN. The saliency DNN produces a saliency map $D_\theta(i, j) \in \mathbb{R}^{H \times W \times 1}$ for each coordinate $1 \leq i \leq H$ and $1 \leq j \leq W$. This score map will then guide the sampling of the input image F using equations 4 and 5, producing $\hat{F}$. The resulting $\hat{F} \in \mathbb{R}^{h \times w \times 3}$ will have an enlarged IOI region to enhance the performance of the instance segmentation operation conducted by the segmentation network $S(.)$, which produce segmentation mask $Y \in \mathbb{R}^{h \times w \times 1}$.

The problem of foveated instance segmentation differs from conventional segmentation in that we only need the segmentation map for the IOI, not the entire image. Therefore, it is unnecessary for the segmentation network $S(.)$ to produce a pixel-wise output for every object within F. Instead, we modify the segmentation network to include two branches, $S = \{S_{\text{seg}}(.), S_{\text{cls}}(.)\}$, as illustrated in Figure 6 (b). The first branch, $S_{\text{seg}}(.)$, produces a binary map $Y_{\text{bm}} \in \mathbb{R}^{h \times w \times 1}$ to represent the IOI mask, where elements of Y_{bm} is set to 1 for regions belonging to the IOI and 0 otherwise. The second branch, $S_{\text{cls}}(.)$, classifies the object within the IOI, yielding an output $Y_{\text{cls}} \in \mathbb{R}^{C \times 1}$, where C is the number of possible classes. The final segmentation label $Y \in \mathbb{R}^{h \times w \times 1}$ is then produced by performing an outer product between Y_{cls} and Y_{bm}, producing the final segmentation mask of $Y_{\text{cm}} \in \mathbb{R}^{h \times w \times C}$ This design reduces the amount of output generated by the segmentation network and simplifies the training process by leveraging the characteristics of the foveated segmentation task.

In practice, S_{seg} can be any pretrained neural network designed for segmentation tasks, such as SegFormer [49], DeepLab [8], among others. During the inference phase, the Y_{cm} is upsampled using the deterministic interpolication process based on the sampler G^h and G^w.

Loss Design To compute the loss, we follow a methodology of [19], where the ground truth mask $Y_{gt} \in \mathbb{R}^{H \times W \times C}$ is subsampled using the identical saliency score map $D_\theta(i, j)$ as the input F, resulting in a subsampled version of the ground truth map $Y_{gt}^{'} \in \mathbb{R}^{h \times w \times C}$. The Dice loss $\mathbb{L}_{dice}$ is then computed between $Y_{gt}^{'}$ and Y_{cm}. Additionally, a unique characteristic of foveal instance segmentation is that the IOI is sometimes quite small (e.g., objects like a knife or pot, as shown in Figure 6 (a)). Consequently, when calculating the pixelwise loss function, if pixels within the IOI are weighted equally with those in the non-IOI region, the results tend to be heavily biased toward the segmentation mask of the non-IOI region, leading to an unintended emphasis on non-IOI regions. To mitigate this, we weigh the pixelwise Focal loss by the inverse of the area of the IOI and the non-IOI region, specifically, the loss $\mathbb{L}_{tot}$ can be defined as:

$$\mathbb{L}_{tot} = L_{dice}(Y_{gt}^{'}, Y_{\text{cm}}) + \lambda L_{focal}(Y_{gt}^{'}, Y_{\text{cm}}) \qquad (6)$$

where, $\mathbb{L}_{dice}(.)$ and $\mathbb{L}_{focal}(.)$ are the dice loss and weighted focal loss functions, respectively. λ denotes the relative importance between $\mathbb{L}_{dice}(.)$ and $\mathbb{L}_{focal}(.)$.

Alternative Training Strategy To train FSNet, we employ an alternate training strategy, as illustrated in Figure 6 (c). Initially, the segmentation neural network components, including $S_{\text{seg}}(.)$ and $S_{\text{cls}}(.)$, are kept frozen while the saliency DNN undergoes training for several epochs. Subsequently, the saliency DNN is frozen, and the segmentation neural network is fine-tuned with a distinct learning rate.

3.3. FovealSeg Framework

Building on the description of FSNet in Section 3.2, this section discusses how it integrates into the FovealSeg framework to enable efficient detection across multiple frames. The FovealSeg algorithm is outlined in Algorithm 1. Initially, a simple criterion is applied to detect the occurrence of a saccade by calculating the difference between the current and previous gaze positions (line 4). If a saccade is detected, instance segmentation for the current frame can be skipped due to the reduced sensitivity of the human visual system during a saccade (line 6). If no saccade occurs, the similarity between the current frame F^t and the initial frame of the current segment F^{init} is assessed by computing their difference. If this difference exceeds a predefined threshold β (line 8), it indicates a significant change in the scene, triggering a full re-execution of instance detection (line 9), and record the updated segmentation mask at the gaze location (line 10). If not, the current gaze location is analyzed to determine if it remains within the region defined by the segmentation mask M_{last}. If it does, M_{last} can be reused; otherwise FSNet must be executed with new gaze location (line 16).

Algorithm 1: FovealSeg Algorithm

```
Input: $T$ is the total time. $F^{init}$ is the initial frame of
       current video segment. $g_{last}$ and $M_{last}$ are
       buffered gaze location and segmentation mask. $g_t$,
       $F^t$ and $M_t$ are current gaze location, input frame
       and segmentation mask. Threshold $\alpha$ and $\beta$ for the
       detection of saccade and end of segment.
1  Initiation
2      $F^{init} = \varnothing$, $g_{last} = \varnothing$, $M_{last} = \varnothing$
3      for $1 \leq t \leq T$ do
4          if $|g_t - g_{last}|^2 > \alpha$ then
5              $g_{last} \leftarrow g_t$;
6              Saccade detect, halt rest operations.
7          else
8              if $\sum_{ij} |F_{ij}^t - F_{ij}^{init}| > \beta$ then
9                  Run FSNet with $F^t$ and $g_t$, get $M^t$;
10                 $F^{init} \leftarrow F^t$, $g_{last} \leftarrow g_t$, $M_{last} \leftarrow M_t$;
11                 return $M_t$
12             else
13                 if $g_t$ is within IOI regions of $M_{last}$ then
14                     return $M_{last}$
15                 else
16                     Run FSNet with $F^t$ and $g_t$, get $M^t$;
17                     $g_{last} \leftarrow g_t$, $M_{last} \leftarrow M_t$;
18                     return $M_t$
```

Dataset	Image size	Number of classes	Size of dataset
Cityscape	512 × 1024	19	20000
ADE20K	640 × 640	31	30000
LVIS	640 × 640	50	100000
Aria Everyday Activities	1402 × 1402	100	60000

Table 1. The overview of evaluation dataset used in our work. In each dataset, we uniformly sample instances to ensure a balanced distribution across categories.

4. Evaluation

To validate the effectiveness of the FSNet and FovealSeg frameworks, we perform extensive evaluations using multiple baseline algorithms and datasets. In Section 4.2, we compare FSNet's performance against different baseline methods across three diverse datasets, emphasizing its advancements. In Section 4.3, we provide quantitative performance results of the FovealSeg framework in AR/VR scenarios, showcasing its suitability for immersive environments. Lastly, we present ablation studies in Section 4.4 to further demonstrate our method's optimality and adaptability across various conditions.

4.1. Experiment settings

Datasets: We utilize four publicly available datasets: Cityscapes [9], ADE20K [54], LVIS [14], and Aria Everyday Activities [31]. Given that these datasets are designed for full-image segmentation, we applied a gaze-

Method	**CityScapes** (64×128)		**ADE20K** (80×80)		**LVIS** (80×80)		**Aria** (180×180)	
	IoU $\uparrow$	**IoU**$'$ $\uparrow$	**IoU** $\uparrow$	**IoU**$'$ $\uparrow$	**IoU** $\uparrow$	**IoU**$'$ $\uparrow$	**IoU** $\uparrow$	**IoU**$'$ $\uparrow$
Avg+DeepLab	0.26	0.27	0.39	0.41	0.35	0.36	0.36	0.37
Avg+HRNet	0.20	0.21	0.43	0.44	0.37	0.38	0.39	0.41
Avg+SegFormer-B4	0.25	0.27	0.37	0.39	0.37	0.38	0.36	0.36
Avg+SegFormer-B5	0.27	0.29	0.41	0.42	0.35	0.37	0.37	0.38
LTD [19]	0.37	0.38	0.41	0.41	0.40	0.41	0.41	0.43
FSNet+DeepLab	**0.52**	**0.53**	0.55	0.56	0.53	0.55	0.54	0.56
FSNet+HRNet	0.47	0.49	**0.56**	0.56	**0.54**	**0.55**	0.57	0.58
FSNet+SegFormer-B4	0.46	0.48	0.54	0.55	0.54	0.56	0.56	0.57
FSNet+SegFormer-B5	0.51	0.52	0.55	**0.57**	0.54	0.55	**0.58**	**0.59**

Table 2. Performance comparison on CityScapes, ADE20K, LVIS and Aria Everyday Activities datasets.

aware masking preprocessing technique to enable foveated instance rendering. Due to the space limit, some of the evaluation results and the details of the preprocessing steps are outlined in the Appendix. For each training and test sample, we randomly select a gaze location within the image and define the IOI based on this gaze location. The details can be found in Table 1.

Model selection: We use a lightweight 3-layer U-Net [40] as the saliency DNN and a pre-trained MobileNetV2 [41] as the classification DNN $S_{cls}(.)$. For $S_{seg}(.)$, we adopt several widely-used architectures, including DeepLabV3 [8], SegFormer [49] and HRNet [45]. We modify the architectures of $S_{seg}(.)$ and $S_{cls}(.)$ to ensure alignment with the dimensions of the downsampled input image F' and the distance map C. Two baselines are developed. In the first baseline, the algorithm uniformly subsamples the input F to produce F' instead of using the saliency DNN. The resulting F' is then processed by the segmentation neural network $S(.)$, with $S_{seg}(.)$ and $S_{cls}(.)$ remaining identical to those in FSNet, following the same training strategy. We also compare FSNet with Learn-to-Downsample (LTD) [19], which uses a learnable downsampling approach combined with an edge-based loss function to perform instance segmentation on the entire frame.

Evaluation metrics: We use IoU and IoU' to evaluate instance segmentation performance over IOI within both the full-resolution test input F and its subsampled version F'. Although LTD performs segmentation across the entire frame, we focus only on the IoU within the IOI where the gaze is directed.

4.2. Evaluation Results of FSNet

Table 2 presents a summary of our results. Among the methods, Avg+DeepLab refers to the uniformly subsampling baseline with a pretrained DeepLab as $S_{seg}(.)$, and similarly for other architectures. Likewise, FSNet+DeepLab represents FSNet with a pretrained DeepLab as $S_{seg}(.)$. For ADE20K and LVIS, the input images are uniformly downsampled to $(80, 80)$, while for Cityscapes and Aria Everyday Activities datasets, the input images are downsampled to $(64, 128)$ and $(180, 180)$, respectively.

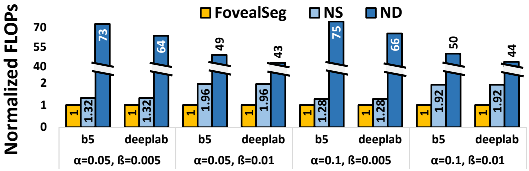


Figure 7. Normalized number of FLOPs of the FovealSeg framework compared to the baselines across different models and combinations of α and β.

As shown in the results, FSNet consistently outperforms the baseline across all datasets and four segmentation backbones, achieving at least a 0.14 improvement in IoU and a 0.15 gain in IoU' compared to the Avg method. For instance, FSNet combined with SegFormer-B5 achieves an IoU of 0.58 and an IoU' of 0.59, surpassing all baselines. Furthermore, FSNet demonstrates strong generalizability across datasets, although slight performance variations are observed due to dataset-specific characteristics. The small gap between IoU and IoU' (< 0.02) suggests that our stage-aware training strategy effectively improves the robustness of both upsampling and downsampling stages.

4.3. Evaluation Results on FovealSeg Framework

The trained FSNet can then be integrated into the FovealSeg framework, as outlined in Algorithm 1, to process segments. To assess the system's performance improvement, we use consecutive frames from the Cityscapes dataset. However, since the Cityscapes dataset lacks gaze location data, we incorporate gaze traces from the Everyday Activity dataset into the Cityscapes dataset. To evaluate FovealSeg, we develop two baseline algorithms. The first baseline algorithm, called **No downsample (ND)**, replaces FSNet with

Two versions are used: SegFormer-B4 and SegFormer-B5

Method	IoU↑	IoU'↑
Avg+DeepLab	0.16	0.17
Avg+HRNet	0.16	0.17
Avg+Seg-B4	0.16	0.18
Avg+Seg-B5	0.16	0.18
FSNet+DeepLab	0.36	0.37
FSNet+HRNet	0.29	0.36
FSNet+Seg-B4	0.32	0.35
FSNet+Seg-B5	0.36	0.38

Table 3. Influence of downsample rate on CityScapes (low resolution 32 × 64).

Method	Kernel Size	IoU ↑	IoU'↑
DeepLab	17	0.48	0.49
HRNet	17	0.45	0.46
Seg-B4	17	0.41	0.44
Seg-B5	17	0.47	0.47
DeepLab	33	0.52	0.53
HRNet	33	0.47	0.49
Seg-B4	33	0.46	0.48
Seg-B5	33	0.51	0.52

Table 4. Effect of gaussian kernel size deployed by sampler on FSNet.

Method	IoU ↑	IoU'↑
DeepLab(w/o)	0.14	0.15
HRNet(w/o)	0.15	0.17
Seg-B4(w/o)	0.15	0.16
Seg-B5(w/o)	0.16	0.16
DeepLab(w/)	0.52	0.53
HRNet(w/)	0.47	0.49
Seg-B4(w/)	0.46	0.48
Seg-B5(w/)	0.51	0.52

Table 5. Influence of gaze information on FSNet performance.

a conventional segmentation DNN that processes the full-resolution input image at a resolution of 640×640. This baseline aims to evaluate the impact of the downsampling operation on computational efficiency. The second baseline applies FovealSeg framework processes images at the downsampled resolution of 64×64 without frame skipping, referred to as **No Skip (NS)**, aim to show the importance of gaze reuse across frames. We configure FovealSeg under different settings for α and β for saccade detection and segment detection, respectively. All the α and β are set to ensure similar frames are grouped and have negligible impact to the model accuracy.

The results in Figure 7, highlight the enhanced computational efficiency achieved by FovealSeg framework compared to the baselines. The high downsampling rate employed by FovealSeg framework reduces the computation required for instance segmentation tasks. As discussed in Section 2.3, leveraging gaze saccades and fixations allows for the elimination of a significant amount of redundant computations, achieving up to a $1.96\times$ reduction in FLOPs. Compared with ND, FovealSeg can achieve up to $75\times$ reduction in computation, underscoring substantial contribution of downsampling to system performance enhancement.

4.4. Ablation Studies

Influence of the Downsample Rate. We start by examining the impact of the downsampling rate in FSNet on performance. Table 3 presents FSNet performance with a downsampled image size F' of 32×64 on the CityScapes dataset. Both IoU and IoU' show a notable decline as the downsampling ratio increases, with the IoU for the DeepLab-based FSNet dropping from 0.52 to 0.36. This trend is consistent across all baseline methods. Nevertheless, even at this low resolution, our FSNet method still outperforms the others.

Influence of the Gaussian Kernel Size in Sampler. We examine how the Gaussian kernel size σ used in equations 4 and 5 affects performance, as it plays a crucial role in the sampling process. Table 4 presents the results on the CityScapes dataset. The results indicate that a larger kernel size yields improved outcomes due to the increased emphasis on the saliency region.

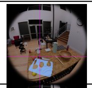
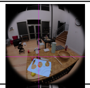
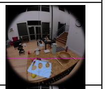
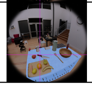
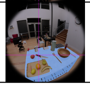
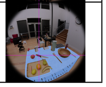
Model	Latency	150	151	152
SegFormer-B5	1860 ms			
FovealSeg	84 ms			

Table 6. Latency and qualitative analysis of sequentially chosen frames (150-152). Due to significant processing latency, SegFormer-B5 may result in a delay between the current gaze location and the predicted segmentation mask.

Influence of the Gaze Information. In FSNet design, we incorporate gaze coordinates (u, v) to guide FSNet in sampling the input image F. In Table 5, we evaluate FSNet's performance on the CityScapes dataset without gaze information by substituting it with random noise. The results show a clear decrease in IoU by over 0.3, highlighting the critical role of gaze location information in FSNet.

4.5. System Performance and Visual Experience

To assess the real-world efficiency of our FovealSeg framework, we compare FSNet+Seg-B5 and SegFormer-B5 to evaluate the speed-accuracy trade-off. To simulate the system performance, we use GPGPU-Sim [20] to run the image segmentation models. GPGPU-Sim [20] is configured to simulate the Jetson Orin NX [10], a widely used edge GPU adopted by ARVR devices [35, 52, 53]. As illustrated in Figure 6, FSNet achieves a latency of 84 ms, making it over **20**× faster than SegFormer-B5, which requires 1860 ms. Notably, FSNet also delivers superior segmentation performance. Its lower latency leads to improved temporal alignment and better visual experience.

5. Conclusion

We present the FovealSeg, which leverages real-time gaze data for instance segmentation focused on the region of IOI. The evaluation results show enhanced performance across various datasets, coupled with notable efficiency improvements, setting the stage for promising future research.

References

[1] NVIDIA Jetson Orin. https://www.siliconhighwaydirect.com/product-p/900-13767-0000-000.htm. 3

[2] Rachel Albert, Anjul Patney, David Luebke, and Joohwan Kim. Latency requirements for foveated rendering in virtual reality. *ACM Trans. Appl. Percept.*, 14(4), 2017. 3

[3] Elena Arabadzhiyska, Okan Tarhan Tursun, Karol Myszkowski, Hans-Peter Seidel, and Piotr Didyk. Saccade landing position prediction for gaze-contingent rendering. *ACM Trans. Graph.*, 36(4), 2017. 2

[4] Elena Arabadzhiyska, Cara Tursun, Hans-Peter Seidel, and Piotr Didyk. Practical saccade prediction for head-mounted displays: Towards a comprehensive model. *ACM Trans. Appl. Percept.*, 20(1), 2023. 2

[5] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017. 1

[6] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. Yolact: Real-time instance segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9157–9166, 2019. 1

[7] Fergus W Campbell and Robert H Wurtz. Saccadic omission: why we do not see a grey-out during a saccadic eye movement. *Vision research*, 18(10):1297–1303, 1978. 2

[8] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation, 2017. 3, 5, 7

[9] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 6

[10] NVIDIA Corporation. Jetson agx orin. Online. Accessed: 2024-11-07. 8

[11] Eye tracking on Meta Quest Pro. https://shorturl.at/WZBzA, 2022. 3

[12] Jasper H Fabius, Alessio Fracasso, Tanja CW Nijboer, and Stefan Van der Stigchel. Time course of spatiotopic updating across saccades. *Proceedings of the National Academy of Sciences*, 116(6):2027–2032, 2019. 2

[13] Antonin Gilles, Pierre Le Gargasson, Grégory Hocquet, and Patrick Gioia. Holographic near-eye display with real-time embedded rendering. In *SIGGRAPH Asia 2023 Conference Papers*, pages 1–10, 2023. 3

[14] Agrim Gupta, Piotr Dollár, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation, 2019. 6

[15] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 1

[16] Tairan He, Zhengyi Luo, Xialin He, Wenli Xiao, Chong Zhang, Weinan Zhang, Kris Kitani, Changliu Liu, and Guanya Shi. Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning. *arXiv preprint arXiv:2406.08858*, 2024. 3

[17] Baosheng James Hou, Yasmeen Abdrabou, Florian Weidner, and Hans Gellersen. Unveiling variations: A comparative study of vr headsets regarding eye tracking volume, gaze accuracy, and precision. In *2024 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, pages 650–655. IEEE, 2024. 3, 4

[18] Saad Idrees, Matthias P Baumann, Felix Franke, Thomas A Münch, and Ziad M Hafed. Perceptual saccadic suppression starts in the retina. *Nature communications*, 11(1):1977, 2020. 2

[19] Chen Jin, Ryutaro Tanno, Thomy Mertzanidou, Eleftheria Panagiotaki, and Daniel C Alexander. Learning to downsample for segmentation of ultra-high resolution images. *arXiv preprint arXiv:2109.11071*, 2021. 4, 5, 6, 7

[20] Mahmoud Khairy, Zhesheng Shen, Tor M. Aamodt, and Timothy G. Rogers. Accel-sim: An extensible simulation framework for validated gpu modeling. *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, pages 473–486, 2018. 8

[21] Mahmoud Khairy, Zhesheng Shen, Tor M Aamodt, and Timothy G Rogers. Accel-sim: An extensible simulation framework for validated gpu modeling. In *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, pages 473–486. IEEE, 2020. 3

[22] Jonghyun Kim, Youngmo Jeong, Michael Stengel, Kaan Akşit, Rachel Albert, Ben Boudaoud, Trey Greer, Joohwan Kim, Ward Lopes, Zander Majercik, Peter Shirley, Josef Spjut, Morgan McGuire, and David Luebke. Foveated ar: dynamically-foveated augmented reality display. *ACM Trans. Graph.*, 38(4), 2019. 2

[23] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023. 1

[24] Eileen Kowler. Eye movements: The past 25 years. *Vision research*, 51(13):1457–1483, 2011. 2

[25] Yuna Kwak, Eric Penner, Xuan Wang, Mohammad R Saeedpour-Parizi, Olivier Mercier, Xiuyun Wu, Scott Murdison, and Phillip Guan. Saccade-contingent rendering. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–9, 2024. 2

[26] Huaijia Lin, Ruizheng Wu, Shu Liu, Jiangbo Lu, and Jiaya Jia. Video instance segmentation with a propose-reduce paradigm. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1739–1748, 2021. 3

[27] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2018. 5

[28] Wenxuan Liu, Budmonde Duinkharjav, Qi Sun, and Sai Qian Zhang. Fovealnet: Advancing ai-driven gaze tracking solutions for efficient foveated rendering in virtual reality. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–11, 2025. 2

[29] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 1

[30] Lester C. Loschky and Gary S. Wolverton. How late can you update gaze-contingent multiresolutional displays without detection? *ACM Trans. Multimedia Comput. Commun. Appl.*, 3(4), 2007. 2

[31] Zhaoyang Lv, Nicholas Charron, Pierre Moulon, Alexander Gamino, Cheng Peng, Chris Sweeney, Edward Miller, Huixuan Tang, Jeff Meissner, Jing Dong, Kiran Somasundaram, Luis Pesqueira, Mark Schwesinger, Omkar Parkhi, Qiao Gu, Renzo De Nardi, Shangyi Cheng, Steve Saarinen, Vijay Baiyya, Yuyang Zou, Richard Newcombe, Jakob Julian Engel, Xiaqing Pan, and Carl Ren. Aria everyday activities dataset, 2024. 3, 6

[32] Dmitrii Marin, Zijian He, Peter Vajda, Priyam Chatterjee, Sam Tsai, Fei Yang, and Yuri Boykov. Efficient segmentation: Learning downsampling near semantic boundaries. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2131–2141, 2019. 4

[33] Ethel Matin. Saccadic suppression: a review and an analysis. *Psychological bulletin*, 81(12):899, 1974. 2

[34] Meta Ray-Ban Series. https://blocksandfiles.com/2023/10/02/meta-ray-ban-emmc-flash/, 2024. 1, 3

[35] Junrui Pan and Timothy G Rogers. Crisp: Concurrent rendering and compute simulation platform for gpus. 3, 8

[36] 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. 2

[37] Frano Rajič, Lei Ke, Yu-Wing Tai, Chi-Keung Tang, Martin Danelljan, and Fisher Yu. Segment anything meets point tracking. *arXiv preprint arXiv:2307.01197*, 2023. 3

[38] Adria Recasens, Petr Kellnhofer, Simon Stent, Wojciech Matusik, and Antonio Torralba. Learning to zoom: a saliency-based sampling layer for neural networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 51–66, 2018. 4

[39] DA Robinson. The mechanics of human saccadic eye movement. *The Journal of physiology*, 174(2):245, 1964. 2

[40] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. 7

[41] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks, 2019. 7

[42] Niklas Stein, Diederick C Niehorster, Tamara Watson, Frank Steinicke, Katharina Rifai, Siegfried Wahl, and Markus Lappe. A comparison of eye tracking latencies among several commercial head-mounted displays. *i-Perception*, 12(1): 2041669520983338, 2021. 3, 4

[43] Chittesh Thavamani, Mengtian Li, Nicolas Cebron, and Deva Ramanan. Fovea: Foveated image magnification for autonomous navigation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 15539–15548, 2021. 4

[44] Chittesh Thavamani, Mengtian Li, Francesco Ferroni, and Deva Ramanan. Learning to zoom and unzoom. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5086–5095, 2023. 4

[45] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep high-resolution representation learning for visual recognition, 2020. 3, 7

[46] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. Solov2: Dynamic and fast instance segmentation. *Advances in Neural information processing systems*, 33:17721–17732, 2020. 1

[47] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8741–8750, 2021. 3

[48] Junfeng Wu, Yi Jiang, Song Bai, Wenqing Zhang, and Xiang Bai. Seqformer: Sequential transformer for video instance segmentation. In *European Conference on Computer Vision*, pages 553–569. Springer, 2022. 3

[49] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in neural information processing systems*, 34: 12077–12090, 2021. 1, 3, 5, 7

[50] Bin Yan, Yi Jiang, Jiannan Wu, Dong Wang, Ping Luo, Zehuan Yuan, and Huchuan Lu. Universal instance perception as object discovery and retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15325–15336, 2023. 3

[51] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5188–5197, 2019. 1, 3

[52] Baoheng Zhang, Yizhao Gao, Jingyuan Li, and Hayden Kwok-Hay So. Co-designing a sub-millisecond latency event-based eye tracking system with submanifold sparse cnn. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5771–5779, 2024. 3, 8

[53] Ziliang Zhang, Zexin Li, Hyoseung Kim, and Cong Liu. Boxr: Body and head motion optimization framework for extended reality. *arXiv preprint arXiv:2410.13084*, 2024. 3, 8

[54] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127(3):302–321, 2019. 6