Segment Only Where You Look: Leveraging Human Gaze Behavior for Efficient Computer Vision Applications in Augmented Reality

Tianhua Xia
Tandon School of Engineering
New York University
New York City, NY, USA
tx856@nyu.edu

Haiyu Wang
Tandon School of Engineering
New York University
New York City, NY, USA
hw3689@nyu.edu

Sai Qian Zhang
Tandon School of Engineering
New York University
New York City, NY, USA
sai.zhang@nyu.edu

Abstract

Augmented reality (AR) comprises groundbreaking technologies that are reshaping the landscape of human interaction. Image segmentation, which divides a user front-scene frame into more manageable parts for analysis, is of paramount importance since this technique enables AR systems to extract digital information precisely from the real world by identifying and isolating specific objects in the user's surroundings. Despite its importance, the segmentation task imposes substantial computational demands and processing delays on AR devices, significantly degrading the user experience.

In this work, we aim to reduce the high computational costs of the segmentation task in AR by leveraging natural human eye dynamics and focusing on segmenting only where you look (SOLO). This involves co-optimizing image segmentation algorithms with underlying hardware for greater efficiency. We introduce SOLO algorithm, an efficient deep learning framework that takes high-resolution input images and user eye images to effectively segment only the instance of interest. Integrated with the saliency-based sensing (SBS) and SOLO accelerator as a plug-in for SoCs of the AR device, SOLO significantly lowers the computational costs for image segmentation, achieving up to a 12× reduction in end-to-end latency compared to other baselines.

CCS Concepts: • **Computing methodologies** → **Image segmentation**; *Interest point and salient region detections*; **Video segmentation**; *Neural networks*; • **Computer systems organization** → **System on a chip**; **Sensors and actuators**; • **Hardware** → *Sensor applications and deployments.*

Keywords: Computer Vision; Video Segmentation; Gaze Tracking; Saccade Detection; Hardware Accelerator; Augmented Reality

ASPLOS '26, Pittsburgh, PA, USA

ACM ISBN 979-8-4007-2359-9/2026/03
https://doi.org/10.1145/3779212.3790216

ACM Reference Format:
Tianhua Xia, Haiyu Wang, and Sai Qian Zhang. 2026. Segment Only Where You Look: Leveraging Human Gaze Behavior for Efficient Computer Vision Applications in Augmented Reality. In *Proceedings of the 31st ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS '26), March 22–26, 2026, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 18 pages. https://doi.org/10.1145/3779212.3790216

1 Introduction

Augmented reality (AR) and virtual reality (VR), collectively referred to as ARVR, represent groundbreaking technologies that are reshaping the landscape of human interaction. Specifically, by merging the digital and physical realms, AR enhances sensory perception and brings digital elements into the real world, enhancing everyday tasks and professional operations with extraction of useful information. The importance of AR extends beyond mere novelty; it holds significant potential for transforming industries such as healthcare [38, 44, 111], manufacturing [15, 78, 93], and education [5, 104, 116], by making abstract concepts tangible.

Computer vision applications are essential in advancing AR systems. Specifically, image segmentation, which divides an image into more manageable parts for analysis, serves as a fundamental application and plays a critical role in the functionality of AR [31, 32, 48, 68, 75]. This technique allows AR systems to accurately extract information from the real world by identifying and isolating specific objects in the user's environment, offering the users the first-hand knowledge of the AR world. Examples are depicted in Figure 1, within an AR grocery application, real-time image segmentation allows the users to identify products on a shelf in real time as they look at them (Figure 1 (a)). Similarly, as shown in Figure 1 (b), in educational settings, segmentation can identify different components of complex diagrams the user is gazing at, enhancing the learning experience for students. The identified objects can then be further processed by the other applications (e.g., vision-language model (VLM)) to generate additional detailed explanation, as shown in Figure 2 (a). Finally, image segmentation allows users to directly edit objects of interest, facilitating a seamless immersive experience between the physical and virtual worlds (Figure 1

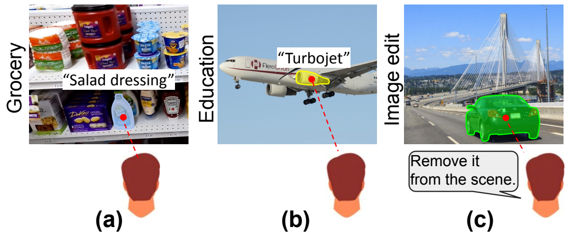

Figure 1. The user wears the AR device and views various scenes, with real-time image segmentation applied.

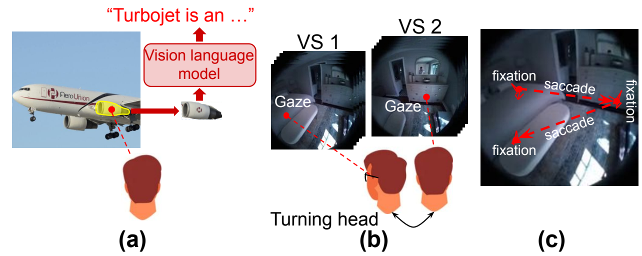

Figure 2. (a) An example use case of segmentation. (b) User gaze pattern and (c) saccadic movements in AR environment.

(c)). Additional applications of image segmentation in AR have been explored in [7, 31, 32, 105].

Despite its significance, the segmentation task presents considerable computational challenges, especially on resource-constrained AR devices, primarily because of the high resolution of images these devices capture. For example, the Meta Aria AR glasses [27] have a high-resolution outer camera recording at 2880 × 2880, while the VIVE Pro 2 [3] features an outer camera with a 2448 × 2448 resolution. This heavy data load leads to significant computational latency, severely limiting performance and responsiveness.

To explore this issue, we evaluate classical computer vision tasks including image segmentation and classification using well-established neural networks, including HRNet [100] and ViT-base [26], with an input resolution of 2880 × 2880. We conduct the experiments on the Jetson Orin NX edge GPU [2] to simulate the AR computation hardware. This setup is often used in prior research to evaluate edge GPU performance in AR devices [30, 37, 83, 92, 129, 136]. Table 1 shows that the average processing latencies are 3347 ms for HRNet and 3942 ms for ViT-base, respectively. These latencies are far from meeting the requirements for a seamless visual experience, as previous research suggests that 50–70 ms is necessary for optimal visual fluidity [6].

In contrast to typical use cases, AR device users exhibit distinct behaviors: they often focus on specific, small areas within a view before shifting their attention to another area. For example, as illustrated in Figure 2 (b), a user wearing AR glasses stands in a bedroom. In the left part of Figure 2 (b), the user focuses on the bed for a few seconds before turning their head to look at the wardrobe, as shown in the right part of Figure 2 (b). In this scenario, the video frames can be segmented into two parts based on the user's head movements. In the first part, where the frames are similar, the user's gaze is mainly on the bed, enabling the reuse of instance segmentation results for the bed across multiple frames. In the second segment, the segmentation can similarly be limited to just the wardrobe. Additionally, as depicted in Figure 1, it is advantageous to segment the instances of interest (IOI) currently under the user's gaze, which often reflects the user's focus and interest at the current moment. This observation presents an inherently efficient method for image segmentation in AR by **concentrating processing on IOI identified by the user's gaze, and ignoring other regions**. This method aligns naturally with the concept of *foveated rendering* [84], which enhances graphical performance by rendering images at full resolution only in the user's direct line of sight, while diminishing detail in the peripheral vision to save on computational effort. In this work, we aim to mitigate the high computational costs associated with image segmentation in AR by leveraging natural human eye dynamics and adopting the principle of *Segmenting Only Where You Look* (SOLO). Our approach involves co-optimizing AI algorithms with underlying hardware to enhance efficiency. We introduce the *SOLO algorithm*, a lightweight deep learning framework that tracks human eye movements and generates saliency maps to guide the outer camera in performing *saliency-based sensing (SBS)* for efficient segmentation. Integrated with the *SOLO accelerator* as a plug-in for AR system-on-chips (SoCs), it significantly reduces sensing, communication, and computation overhead, leading to substantial speedup and energy saving. Our key contributions are as follows:

- We introduce a deep learning framework known as *SOLONet*, that takes high-resolution input images and eye images to effectively segment only the IOI. SOLONet is designed to be integrated with any existing segmentation network, greatly enhancing its generalizability.
- We propose a novel image sensing mechanism called *Saliency-based Sensing (SBS)*, which substantially reduces sensing costs while significantly lowering the energy required for sensing and transmission.
- We propose the *SOLO Accelerator* as a plug-in for AR SoC. This accelerator processes captured eye images and generates a saliency map to initiate SBS, significantly facilitating the segmentation process on GPU.
- Evaluation results show that SOLO achieves an 8.6× speedup and 9.1× energy savings compared to other baseline hardware platforms, while maintaining a negligible impact on segmentation accuracy.

Input size	160 × 160	320 × 320	640 × 640	1440 × 1440	**2880 × 2880**
HRNet [113]	42 ms	96 ms	423 ms	852 ms	**3347 ms**
ViT-B [26]	67 ms	163 ms	495 ms	1016 ms	**3942 ms**

Table 1. Processing latencies under different resolutions.

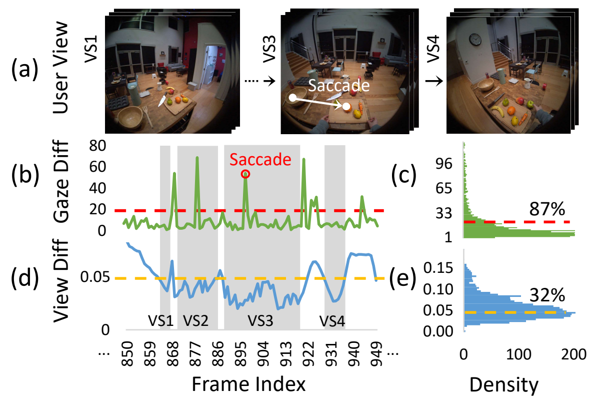

Figure 3. User gaze study on Aria everyday dataset.

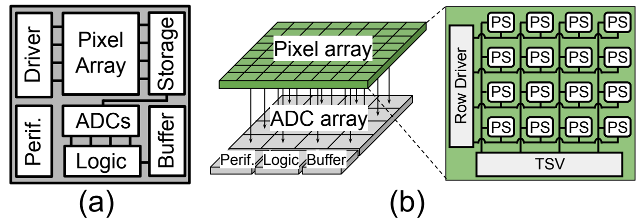

Figure 4. (a) 2D and (b) 3D image sensor layout.

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 (c). During the fixation stage, human gaze remains focused around a single point, with visual acuity mostly concentrated in the nearby region. In addition to fixation, human eye engages in saccadic motion, where it rapidly jumps from one focal point to another. During a saccade, the visual system's sensitivity temporarily diminishes, referred to as *saccadic suppression* [49, 72]. This sensitivity change helps prevent the brain from perceiving rapid, blurred movement of the visual field as eyes quickly shift focus. The saccade duration ranges from 30 ms to 250 ms depending on the gaze movement distance [13]. It takes 50 ms to recover the sensitivity after the saccade ends [25, 77].

2.2 Human Behavior in AR Environment

To further study the human gaze behavior while using AR devices, we conduct an extensive analysis using the Aria Everyday Activities Dataset [67]. This dataset comprises 143 sequences of frames captured by the AR device, aligning with the user's field of view and tracking gaze movements across each frame in various environments, as shown in Figure 3.

As the user changes their head orientation, the front view changes accordingly. To quantify this head movement, we calculate the image difference by measuring the Euclidean distance between corresponding pixels of consecutive frames. Minimal pixel differences between frames suggest that the user is keeping a steady head orientation on a static front scene over time. If the pixel difference is below this threshold, the frames are considered highly similar and virtually indistinguishable by the human eye. These frames can be grouped into a single *video segment (VS)*, as indicated by Figure 2 (b) and Figure 3 (a). Figure 3 (d) shows the percentages of pixel difference over a sample time interval spanning four VSs, defined by grouping consecutive frames where the percentage of pixel changes is below a threshold, indicated by the yellow line. As shown in Figure 3 (e), 32% of consecutive frames have less than 5% pixel value changes, showing a high potential for reusing segmentation results across frames.

Furthermore, within each VS, the segmentation results can be reused if the gaze remains relatively stable, consistently pointing to the same IOI. To support this observation, we analyze the distances (in pixel) between consecutive gaze locations within a VS, as depicted in Figure 3 (b). Our analysis establishes that a threshold of 20 pixels effectively groups gaze locations during the fixation phase, where user looks at only one IOI; values exceeding this threshold indicate a saccade, as observed rapid gaze changes in VS3 of Figure 3 (a). As shown in Figure 3 (c), 87% of the frames in each VS have a gaze distance of less than 20 pixels, and AR users typically focus on just one or two IOIs within each VS. This evidence offers an opportunity to improve image segmentation efficiency by **focusing processing on the IOI and reusing segmentation results when gaze shifts are minimal**.

2.3 Image Sensor Architecture

The image sensor is a key component of AR devices, with its architecture shown in Figure 4 (a). It consists of a pixel array, ADCs, readout circuits, and both analog and digital buffers. In operation, photodiodes in the pixel array convert incoming light into analog electrical signals, which are temporarily stored in the analog buffers. Following the readout strategy, the pixel array driver routes these signals to the ADCs for conversion into digital form. Given the large number of pixel cells, it is common for the outputs of multiple photodiodes to share a single ADC [14, 53, 130]. Each group of photodiodes is referred to as a Pixel Sub-array (PS) in Figure 4 (b).

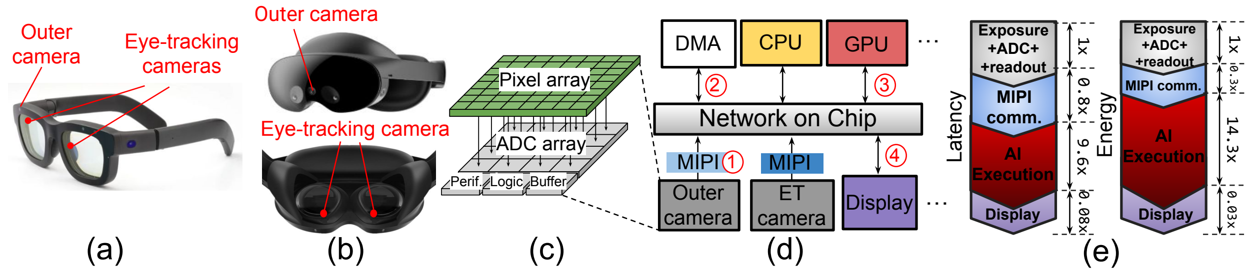

Figure 5. (a) Meta Aria AR Glass. (b) The front and inner view of Meta Quest Pro Head-mounted display (HMD). (c) Architecture of a typical image sensor, which is deployed by both outer camera and ET camera. (d) The architecture of AR system. (e) Latency and energy breakdown o AR system, with values normalized to the sensing component (Plots are not shown to scale).

The digitized data from ADC may then be processed by the image signal processor (ISP) before being stored in digital buffers and transmitted to DRAM via Mobile Industry Processor Interface (MIPI) [4]. A common design is the rolling-shutter image sensor, where each PS row is exposed and read out sequentially. This requires multiple rounds of processing, causing total latency and energy consumption to grow proportionally with the number of pixels captured. Traditional 2D image sensors, as illustrated in Figure 4 (a), place all hardware components on a single layer. This compact layout, constrained by limited physical space, increases routing complexity, which in turn slows data transfer and increases latency. To address these limitations, 3D-stacked image sensors have been developed. As shown in Figure 4 (b), the sensing and processing components are placed on separate layers and connected with Through-silicon vias (TSV). This design accelerates data transfer and reduces routing complexity, making 3D-stacked sensors increasingly popular in AR/VR devices [18, 39, 41, 60–62, 81, 98, 110, 127].

Among these stages, ADC+readout and MIPI communication are the important contributors to image sensing latency and energy consumption [22, 46, 96], where ADC+readout includes ADC conversion and subsequent operations prior to MIPI transfer, such as buffer access and data control [22, 96]. The costs of both ADC+readout and MIPI scale proportionally with the number of pixels. For example, Choi et al. [22] report that ADC+readout accounts for 94% of the total power consumption of the image sensor, whereas pixel exposure contributes only 4%, excluding the MIPI component. In terms of latency, exposure can reach the millisecond range in certain cases, particularly under low-light conditions, and may dominate the overall sensing latency [56, 71, 126]. By contrast, ADC+readout for a single pixel row typically completes within tens of microseconds, and the total image latency scales with resolution. Overall, the exact timing is highly dependent on sensor architecture (e.g., the number of ADCs) and the application context [23, 61, 62]. For example, Seo et al. [96] report exposure and ADC+readout times of 45 µs and 833 µs, respectively, indicating that ADC+readout accounts for the majority of the total sensing cost.

2.4 AR System Architecture Overview

Figure 5 (a) and (b) showcase the Meta Aria AR glasses [27] and Meta Quest Pro [42]. Each device is equipped with an outer camera that continuously captures the user's front view, producing high-resolution images (e.g., 2880×2880), and an inward-facing eye-tracking (ET) camera that captures monochrome images of the user's eyes in lower resolution. Although the Meta Quest Pro is marketed as a VR device, it can also simulate an AR effect by displaying the front view to the user. Figure 5 (d) illustrates the architecture of the AR SoC, which primarily includes mobile CPU and GPU. The GPU is mainly used for computationally intensive tasks like image rendering and AI workloads. The architecture of a typical camera sensor, used by both the outer camera and the ET camera, is shown in Figure 5 (c).

The numbers in red circles in Figure 5 (d) represent the computational flow of traditional image segmentation tasks on full resolution inputs. The full resolution image is first captured by the outer camera (Step 1) and sent to the DRAM via MIPI (Step 2). The GPU then executes the segmentation model over the full resolution input (Step 3) and generates the mask. To assess the hardware cost of running AI applications on a mobile GPU and how performance changes with input image size, Table 1 demonstrates that reducing image size significantly lowers processing latency for AI tasks. Specifically, a 160×160 input results in 42 ms segmentation latency on HRNet, 80× faster than a 2880×2880 resolution. Finally, the generated mask can then be used to extract the gazed object from the scene for downstream applications and, optionally, overlay it onto the front scene for display with minimal rendering and display costs.

2.5 Gaze Tracking Algorithms

Gaze tracking methods are commonly divided into model-based and appearance-based approaches [34, 133]. Model-based techniques use a 3D eye model to simulate physiological structures and estimate gaze direction [65, 102, 114, 117]. Typically, these methods involve two phases: (1) generating an eye segmentation label map and fitting a geometric eye model using an eye feature extraction neural network, and (2) calculating the gaze direction from the fitted model [28, 29, 52, 125, 132]. On the other hand, appearance-based methods analyze direct images of the eye to learn a mapping to gaze direction [9, 34, 63, 66, 73, 99, 99, 118, 134, 135]. These approaches often demand more extensive training data compared to model-based techniques.

2.6 Image and Video Segmentation

Semantic segmentation [10, 17, 19, 21, 54, 76, 97, 109, 112, 121] is a fundamental task in computer vision that involves dividing an image into distinct regions or segments, which simplifies the analysis and interpretation of the content. A more advanced approach within this domain is instance segmentation [16, 36, 55, 79], where the goal is to distinguish between individual instances of the same object class. SOLO leverages user gaze input in combination with the captured image to **segment IOI exclusively**.

To make the image segmentation more efficient, previous research has focused on creating learnable input downsampling methods to adjust sampling resolution selectively. In [90], 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 [45, 70, 106], apply similar concepts by learning a saliency score for each pixel to guide the downsampling process, resulting in improved performance for semantic segmentation tasks. In this work, we introduce ESNet, as detailed in Section 3.2, which adaptively downsamples the input image based on the predicted user gaze location and the shape of IOI. Additionally, previous studies [57, 89, 115, 119, 122, 123] on video instance segmentation process consecutive frames together (e.g., VisTR [122] and SeqFormer [119] process 5 consecutive frames at once). This approach utilizes temporal correlations across frames to improve performance; however, it introduces great latency, as processing can only start once all frames are available.

3 SOLO Algorithm

As indicated in Table 1, reducing the size of the captured frame is an effective strategy to decrease the overall processing latency and energy consumption. However, a straightforward approach of averaging and downsampling the input frame will inevitably lead to a degradation in segmentation quality, as the IOI will also be downscaled, making it much harder to identify and analyze.

To tackle this challenge, we propose the SOLONet, an efficient deep neural network that selectively downsamples the captured high-resolution input frame based on the predicted user gaze direction. SOLONet preserves the resolution of the region containing the IOI while intelligently reducing the resolution of less relevant areas. This strategy ensures that critical regions retain their quality while optimizing computational resources, significantly reducing the overall input size. The architecture of SOLONet is illustrated in Figure 6 (a). Upon receiving a high-resolution input image I_f and an eye image I_e, I_f is first evenly subsampled, generating I_f^d. Both I_f^d and I_e are then fed into ESNet (Section 3.2), which produces a saliency score map to guide the SBS process for I_f based on the user's gaze direction, as determined by I_e. The subsampled input frame I_f^s based on saliency score is then passed to the segmentation network S_{seg} for segmentation and classification tasks (Section 3.3). The resulting segmentation label map Y_{cm}^d is then upsampled to match the original scale of I_f. Next, we describe each part in detail.

3.1 Preliminary: Learn to Downsample

Image downsampling can be interpreted as a process that reduces the original image $I_f \in \mathbb{R}^{H\times W\times C}$ to a smaller version $I_f^s \in \mathbb{R}^{h\times w\times C}$, where $h \leq H$ and $w \leq W$. This process is governed by two mapping functions, $g^1(.)$ and $g^2(.)$, which map the 2D coordinate (i, j) of the downsampled image I_f^s to the coordinate $(g^1(i, j), g^2(i, j))$ in the original image I_f. Consequently, each pixel in I_f^s can be expressed as:

$$I_f^s[i, j] = I_f[g^1(i, j), g^2(i, j)] \tag{1}$$

where $I_f^s[i, j]$ represents the value of the element at coordinate (i, j) within I_f^s. This mapping function can define all types of downsampling operations.

The mapping functions $g^1(.)$ and $g^2(.)$ can be made learnable [45, 107] using a saliency score map $S \in \mathbb{R}^{h\times w\times 1}$, which indicates the relative sampling density at each pixel location (i, j). Specifically, the mapping function can be defined as:

$$g^1(i, j) = \frac{\sum_{i',j'} S(i', j')k_\sigma((i/h, j/w), (i'/H, j'/W))i'}{\sum_{i',j'} S(i', j')k_\sigma((i/h, j/w), (i'/H, j'/W))} \tag{2}$$

$$g^2(i, j) = \frac{\sum_{i',j'} S(i', j')k_\sigma((i/h, j/w), (i'/H, j'/W))j'}{\sum_{i',j'} S(i', j')k_\sigma((i/h, j/w), (i'/H, j'/W))} \tag{3}$$

where $k_\sigma(.,.)$ is a Gaussian kernel with a standard deviation of σ. Consequently, the downsampling process can be made learnable by optimizing the saliency score map $S(.)$ for optimal task performance. To restore I_f^s to its original size, a reverse sampler g^{-1} maps the downsampled image back to the original space utilizing an interpolation function. $g^1(i, j)$ and $g^2(i, j)$ denote the x and y 2D mapped coordinates in I_f^s that correspond to the (i, j) coordinate in I_f. Next we discuss each component of SOLONet in detail.

3.2 ESNet

ESNet processes the eye image I_e, which contains the user's gaze information, along with a uniformly subsampled version $I_f^d \in \mathbb{R}^{h\times w\times 3}$ of the input frame $I_f \in \mathbb{R}^{H\times W\times 3}$, to generate the index map H. In addition, ESNet is also responsible for detecting the occurrence of the saccade. The architecture of ESNet is shown in Figure 6 (b). The eye image I_e is

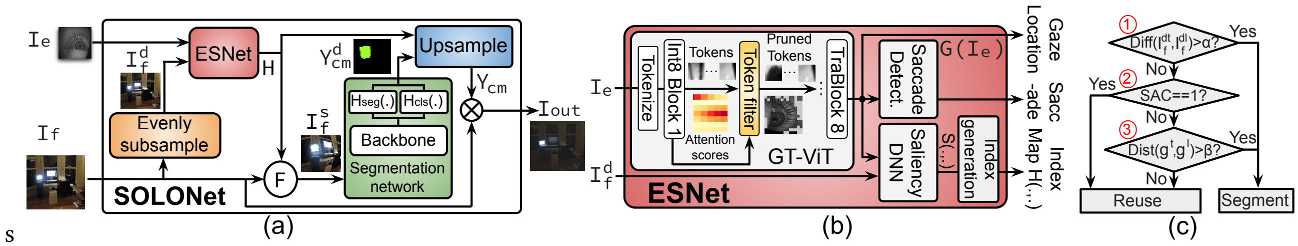

Figure 6. (a) SOLONet architecture. (b) Design of the ESNet. (c) Logic flowchart of the SOLO Streaming Algorithm.

first passed to the Gaze Tracking Vision Transformer (GT-ViT), denoted as $G(.)$ which consists of eight transformer blocks. Each block includes six heads with an embedding dimension of 384. A linear layer is attached at the end of the ViT to produce the 2D gaze direction $G(I_e)$. To reduce the computational cost of the ViT, unnecessary tokens within the intermediate results are pruned based on their attention scores. Tokens with an attention score below a predefined threshold are removed. Finally, all elements within the activation and weight matrices are quantized to 8 bits to further enhance the efficiency. The predicted gaze $G(I_e)$ from the GT-ViT is then used by the saccade detection module, which consists of a single-layer recurrent neural network (RNN) to produce a binary output indicating the presence of a saccade. Given that the visual system's sensitivity temporarily diminishes during a saccade, as discussed in Section 2.1, the occurrence of a saccade can be used to selectively skip the segmentation operations, as explained in Section 3.5. Additionally, the predicted gaze direction $G(I_e)$, along with the subsampled version I_f^d of the image, is also used to produce the saliency score map S. The score map is then used to guide the sampling of the input image I_f with Equations 2 and 3, producing index maps H, which are then subsampled over I_f to generate I_f^s (Figure 6 (a)). This is then passed to gaze-aware segmentation network detailed in Section 3.3.

3.3 Gaze-aware Segmentation Network

The problem of instance segmentation tackled by SOLO differs from conventional segmentation in that we only need to segment IOI, rather than the entire image. Therefore, it is unnecessary for the segmentation network to produce a pixel-wise output for every object within I_f. Instead, we modify the segmentation network S_{seg} to add two neural network heads, $H = \{H_{\text{seg}}(.), H_{\text{cls}}(.)\}$, to the segmentation backbone S_{seg}, as shown in Figure 6 (a). The first head, $H_{\text{seg}}(.)$, produces a binary label map $Y_{\text{bm}} \in \mathbb{R}^{h\times w\times 1}$, where a value of 1 indicates the region of IOI, and 0 indicates the areas outside of it. The second head, $H_{\text{cls}}(.)$, predicts the class of IOI, yielding an output $Y_{\text{cls}} \in \mathbb{R}^{C\times 1}$, where C is the number of possible object classes. To denote the background region, we introduce an additional class label, resulting in a total of $C+1$ classes. The segmentation label map $Y_{\text{cm}}^d \in \mathbb{R}^{h\times w\times (C+1)}$ is then produced by performing an outer product between Y_{cls} and Y_{bm}. This design on producing the final label map greatly simplifies the training process. Y_{cm}^d is then upsampled using the reverse sampler with interpolation, generating the final segmentation label map Y_{cm}.

3.4 SOLONet Training Methodology

SOLONet is trained in an end-to-end fashion by jointly optimizing ESNet and the gaze-aware segmentation network. Before the training starts, the GT-ViT within the ESNet is first pretrained using the gaze tracking dataset (e.g., OpenEDS2020 [82]) to accelerate the convergence speed of SOLONet. To compute the loss, we follow a methodology of [45], where the ground truth segmentation label map $Y_{\text{cm}}^{\text{gt}} \in \mathbb{R}^{H\times W\times (C+1)}$ and binary label map $Y_{\text{bm}}^{\text{gt}} \in \mathbb{R}^{H\times W\times 1}$ are subsampled with the identical saliency score map as the input I_f using the process described in Section 3.1, resulting in subsampled versions $Y_{\text{cm}}^{\text{s,gt}} \in \mathbb{R}^{h\times w\times (C+1)}$ and $Y_{\text{bm}}^{\text{s,gt}} \in \mathbb{R}^{h\times w\times 1}$. To mitigate the highly imbalanced areas between the IOI and background regions, we use Dice loss [58] to encourage SOLONet to focus more on the segmentation of the IOI rather than the background. Additionally, we introduce a regularization term to encourage ESNet to increase the saliency score within the region of IOI. This can be achieved by minimizing the l_2 distance between ground-truth binary label map $Y_{\text{bm}}^{\text{s,gt}}$ and the saliency score map S. In summary, the loss $\mathbb{L}_{tot}$ can be defined as:

$$\mathbb{L}_{tot} = \mathbb{L}_{Dice}(Y_{cm}^d, Y_{\text{cm}}^{\text{s,gt}}) + \lambda \mathbb{L}_{mse}(Y_{\text{bm}}^{\text{s,gt}}, S) \tag{4}$$

where $\mathbb{L}_{Dice}(.)$, $\mathbb{L}_{mse}(.)$ represent the Dice loss and l_2 distance loss function, respectively. λ denotes the relative importance between $\mathbb{L}_{Dice}(.)$ and $\mathbb{L}_{mse}(.)$.

3.5 SOLO Streaming Algorithm

Building on the AR user gaze behavior described in Section 2.2, this section discusses how it integrates into the SOLO Streaming Algorithm (SSA) to enable efficient segmentation of consecutive video frames. During the real-time operation, SOLONet is triggered only when the masked region of I_f^d captured by the front camera undergoes a noticeable change or when there is a shift in the human gaze direction. The logic flowchart of SSA is illustrated in Figure 6 (c), a simple criterion is applied to detect if the view

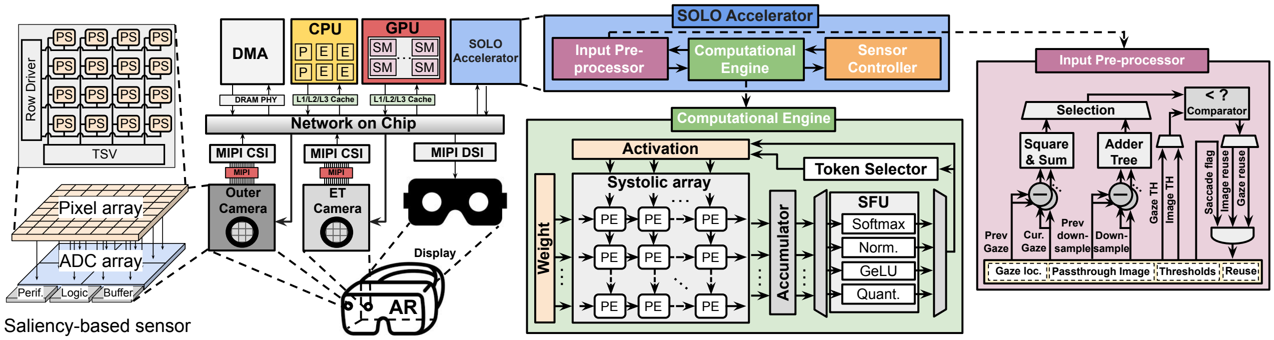

Figure 7. An overview of the AR system: The SOLO accelerator is designed to integrate with the AR SoC to accelerate the segmentation process. Additionally, the outer camera is enhanced to support SBS.

is still in the previous segment by calculating the difference between I_f^{dt} and I_f^{dl} (Condition 1), where I_f^{dt} and I_f^{dl} denote the uniformly subsampled versions of the current and last frames, respectively. If this difference exceeds a predefined threshold α, it indicates a significant change in the front scene, triggering a full re-execution of SOLONet. Next, if a saccade is detected by ESNet (Condition 2), instance segmentation for the current frame can be skipped and previous results can be reused due to the reduced sensitivity of the human visual system during a saccade. If not, the current gaze location (g^t) is analyzed to determine if it remains within the same IOI as that (g^l) of the last frame (Condition 3). If a predefined threshold β is exceeded for the gaze location difference, SOLONet must be executed with the new gaze location, otherwise the last segmentation label map can be reused.

4 SOLO Accelerator

Figure 7 presents an overview of the AR system, where the SOLO accelerator, which implements ESNet as shown in Figure 6 (b), functions as a plug-in for the AR device's SoC. We also introduce the *saliency-based sensor*, capable of performing SBS using the output from the SOLO accelerator, significantly reducing sensing and communication latency while minimizing power consumption.

The numbers in blue circles in Figure 8 (a) illustrate the computational flow of SOLO in the AR SoC. The SBS-enabled outer camera first captures the input frame I_f and reads out the evenly subsampled version I_f^d of I_f (Step 1). The eye-tracking camera will capture the user's eye image I_e in parallel (Step 2). Then, I_f^d and I_e are transmitted through MIPI and stored in the DRAM (Step 3). Next, the SOLO accelerator predicts the gaze direction $G(I_e)$, generates the saliency score map S, and checks the segmentation reuse conditions. If a new segmentation label map is required, it initiates the SBS process (Step 4) and drives the SBS-enabled outer camera to re-sense the input image, producing I_f^s (Step 5). I_f^s is then sent to the DRAM via MIPI (Step 6) and processed by the GPU

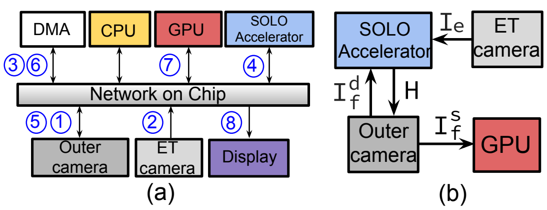

Figure 8. (a) Computational flow of AR SoC. (b) SBS steps.

for the segmentation operation (Step 7). The segmentation label map is then sent to the AR display (Step 8).

The detailed SBS process is depicted in Figure 8 (b). The SOLO accelerator receives I_f^d from the outer camera and I_e from the eye-tracking camera to generate the saliency score map S. The sensor controller in the SOLO accelerator then uses the saliency score map S and creates the index map H with equations 2 and 3, where $H(i,j) = [g^1(i,j), g^2(i,j)]$. The index map H is subsequently transferred to the outer camera, which performs SBS by reading only the corresponding pixels, forming I_f^s. Section 4.1 details the design of saliency-based sensor. The SOLO accelerator is introduced in Section 4.2. Section 4.3 describes the AR SoC timing analysis.

4.1 Saliency-based Sensor Design

The column-parallel ADC architecture, widely used in 3D image sensors [8, 40, 47], increases frame rate by enabling parallel processing of one or more pixel sub-array (PS) rows (typically 4–8) per sensing round [94, 96, 103], thereby improving image sensing performance. However, many sensing rounds are still required to process the entire image, resulting in significant latency and energy consumption. In SOLONet (Section 3), only a subset of pixels is needed for segmentation. To exploit this, we propose *Saliency-Based Sensing* (SBS), which selectively reads out only the required pixels. Unlike prior in-sensor compression approaches [50, 51, 131, 138] that compress data after all pixels have been exposed, and digitized, SBS directly reduces the number of analog signals passed to the ADC. This substantially lowers ADC+readout cost while keeping MIPI transfer overhead minimal.

Specifically, we propose a novel image sensor design for the outer camera in the AR device that supports the SBS defined by the index map H. A simple design featuring an $r \times r$ PS array is shown in Figure 9 (a), where r is set to 4. An ADC array of size $k \times r$ is positioned below the PS array, with each ADC controlling the analog signal conversion of $\frac{r}{k}$ PSs in one column, where k is set to 1 in the example of Figure 9 (a). The detailed PS assignment at each ADC is specified in Figure 9 (b). The SBS procedure is shown in Figure 9 (d) and (e). Assume an index map $H = [(1,2), (1,3), (2,1), (2,2), (2,3)]$ is returned by the SOLO accelerator to sample a 4×4 input frame I_f. The row driver then activates the PSs corresponding to the selected pixels, allowing them to output their stored analog signals to the corresponding ADCs, which convert them into digital values and store them in the buffer. Different from traditional image sensor design where all the pixels in one row are readout by ADC in parallel, in SBS, only the selected pixels will be readout in a row. Figure 9 (d) and (e) show the ADC+readout of the first and second PS row, respectively.

A more detailed design is depicted in Figure 10. The pixel array size is 1440×1440, where 2×2 photodiodes are combined in one PS, resulting in a 720×720 PS array shown in Figure 10 (a), consistent with prior 3D image sensor designs [94, 96, 108]. Within each PS column, those belonging to the same ADC sub-group are connected via one vertical data wire. For example, the PS at row 1, column 1, denoted as (1, 1), and the PS at (716, 1) belong to the same sub-group and are linked by a single light-red vertical wire. Consequently, the number of vertical wires alongside each PS column corresponds to the number of sub-groups associated with that column. Most 3D column-parallel ADC image sensors support 4–8 vertical wires per column [94, 96, 103]. In our design, to maximize parallelism while avoiding excessive routing overhead, the PSs in each column are divided into four interleaved sub-groups. Each ADC sub-group spans PSs in one column. For a PS array of size 720×720, this configuration results in a total of $720 \times 4 = 2880$ ADCs, as shown in Figure 10 (b). Thanks to the 3D architecture, this design greatly simplifies the interconnection between each ADC and the PSs within its sub-group.

When reading analog values from the photodiodes to the ADC, only a subset of PSs forward their sensed inputs to the ADC. This is achieved by activating the corresponding connections to the ADC in each sensing round, thereby greatly reducing both latency and energy consumption.

4.2 SOLO Accelerator

In this section, we describe the hardware design of the SOLO accelerator for efficient ESNet implementation. It is composed of three major components: the input pre-processor, the computational engine, and the sensor controller. The computational engine, as illustrated in Figure 7, is further divided into a systolic array core, a special function unit (SFU), and a token selector.

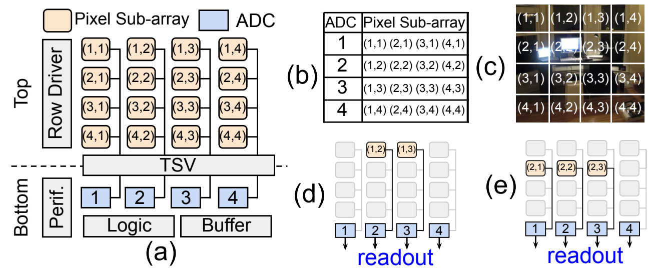

Figure 9. (a) An example of saliency-based sensor with 4×4 PS array (yellow boxes) and 1×4 ADC array (blue boxes). (b) Mapping between ADC and pixel sub-array. (c) An example of 4×4 image. (d) and (e) shows ADC+readout of first and second PS row.

The computational engine consists of a 16×16 2D systolic array that processes inputs in a staggered manner, sending the computed partial sums to the accumulator and SFUs. It utilizes a weight-stationary data flow, where weights are preloaded from the weight SRAM into the registers within each processing element (PE) of the systolic array, as illustrated in Figure 7, while inputs are streamed into the array sequentially. Each PE includes an 8-bit multiply-accumulator (MAC). The SFU handles all nonlinear operations within the ESNet, including activation functions, softmax, normalization, positional embeddings, quantization, as well as the index map generation. The systolic array executes all the GEMM operations within GT-ViT and RNN for saccade detection.

The token selector is responsible for tokenwise pruning supported by GT-ViT. It dynamically computes the importance of each token by summing the attention scores to derive the importance score for the tokens using an adder array. It then prunes tokens with importance scores lower than a predefined threshold, excluding them from subsequent computations. The token pruning method effectively reduces the computational cost of the gaze prediction process.

The input pre-processor, shown as the purple block in Figure 7, is responsible for validating the result reuse condition outlined in Section 3.5. It computes the pixel-wise difference between I_f^{dt} and I_f^{dl}, then sums the values using an adder tree. To determine the gaze difference, the pre-processor calculates the squared distance between gaze locations g^t and g^l. The computed image difference and gaze difference are then compared against the thresholds α and β, respectively. Finally, the comparator results are aggregated with the saccade detection signal to generate the result reuse decision.

The sensor controller triggers the Gaussian kernel convolution on the computational engine to create the index map H from the saliency score map S using Equations 2 and 3, where $H(i,j) = [g^1(i,j), g^2(i,j)]$.

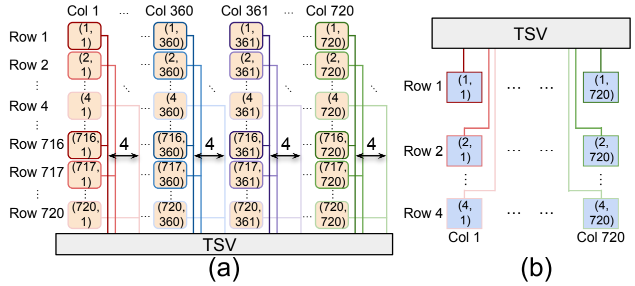

Figure 10. SBS sensor with (a) a 720×720 PS array and (b) a 4×720 ADC array. PSs within the same sub-group share an ADC, indicated by matching border and wire colors.

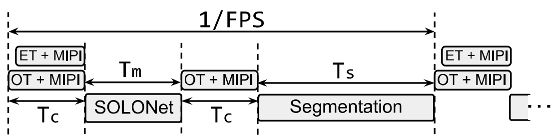

Figure 11. Timing diagram of SOLO execution. ET and OT refer to eye track camera and outer camera sensing latency.

4.3 Computational Flow of SOLO

In this section, we provide a detailed timing analysis of the SOLO algorithm presented in Section 3. The timing diagram is shown in Figure 11. Specifically, let T_c, T_m, and T_s denote the outer image sensing latency for I_f^d and I_f^s, together with the MIPI transfer latency, the SOLONet processing time, and the instance segmentation time, respectively. The total latency to generate the label map Y_{cm} can be estimated as $T_{standard} = 2T_c + T_m + T_s$, which results in a throughput in frames per second (FPS) given by $\frac{1}{T_{standard}}$. Moreover, as illustrated in Section 3.5, the segmentation process for certain frames can be skipped by reusing results from previous frames. The occurrence of such situations depends on the user's eye and head movement behavior. Specifically, let P_{nv}, P_{sac}, and P_{ng} represent the probabilities that the input frame changes (Condition 1 in Figure 6 (c)), the occurrence of a saccade (Condition 2), and the user's gaze location varies (Condition 3), respectively. The probability P_{skip} that the segmentation process is skipped for the current frame is given by:

$$P_{skip} = (1 - P_{nv})P_{sac} + (1 - P_{nv})(1 - P_{sac})(1 - P_{ng}) \quad (5)$$

To detect whether the current frame can be skipped or not, the outer camera needs to sense the current input image I_f and transfer the evenly downsampled version I_f^d of I_f for gaze detection, whose latency can be estimated as $T_{skip} = T_C + T_m$. Considering the probability of frame skipping, the average latency T_{solo} for SOLO to process one input frame can be estimated as:

$$T_{solo} = T_{standard}(1 - P_{skip}) + T_{skip}P_{skip} \quad (6)$$

5 SOLO Accuracy Evaluation

In this section, we evaluate the accuracy performance of SOLO on instance segmentation task on three datasets: LVIS [33], ADE20K [139], and Aria Everyday [27]. The input sizes for each dataset are set to 640×640, 512×512, and 960×960, respectively. For each sample in the training dataset, we randomly select an IOI within the image and use the corresponding ground truth label map of IOI for training.

For the segmentation network S_{seg}, the segmentation head $H_{\text{seg}}(.)$ and classification head $H_{\text{cls}}(.)$ are each composed of three convolutional layers. The backbone of the segmentation network is built using several pretrained DNNs, including HRNet-W32 [113], SegFormer-B1 [120], and DeepLabV3-ResNet101 [17]. We refer to the SOLONet that uses the corresponding backbone network as *HR*, *SF*, and *DL*, respectively. The standard deviation σ of Gaussian kernel $k_\sigma(x, x')$ in Equations 2 and 3 is set to 45, 35, and 50 for LVIS, ADE, and Aria Everyday dataset, respectively. α and β in SSA (Section 3.5) are set to 0.05 and 20, respectively. λ in Equation 4 is set to 0.1. The gaze ViT in the ESNet has 8 layers and 30% of the tokens are pruned.

To evaluate the accuracy of SOLONet, two baseline algorithms are developed. The first baseline algorithm, termed *Average Downsampling* (AD), generates I_f^s by average downsampling over the original I_f. The second baseline, termed *Learn to Downsample* (LTD), as outlined in [45], performs saliency-based downsampling over I_f without considering the gaze location. Additionally, we compare the performance against the original HRNet, Deeplab, and Segformer, which perform conventional image segmentation on the full resolution input and then select the mask of IOI from that of the entire image. This baseline is referred to as *Full Resolution* (FR).

To evaluate the segmentation performance, we adopt *Intersection over Union (IoU)*, a metric used to measure the overlap between the predicted segmentation label map and the ground truth label map. Specifically, we use two types of IoUs: *binary IoU (b-IOU)*, which is calculated using the binary label map Y_{bm} to evaluate segmentation accuracy for the IOI without considering its class label, and *classified IoU (c-IOU)*, which is calculated using Y_{cm} and takes the classification performance of the IOI into account.

5.1 Segmentation Accuracy

We evaluate the segmentation performance of SOLONet. For each selection of dataset and segmentation backbone, we control the computational cost in FLOPs to be roughly the same across different baselines. For the LTD and SOLO baselines, the input image is downsampled to the sizes of 80×80, 64×64, and 120×120 for LVIS, ADE20K, and Aria Everyday, respectively. HR downsamples Aria image to 100×100. For AD, the input image is downsampled to the sizes of 85×85, 70×70, and 125×125 for LVIS, ADE20K, and Aria Everyday, respectively. As shown in Table 2, SOLO consistently

Model	Dataset	AD	LTD	SOLO	GFLOPs	FR	GFLOPs
HR	LVIS	0.5/0.43	0.56/0.49	**0.66/0.56**	**12**	0.53/0.45	516
	ADE	0.48/0.37	0.53/0.43	**0.64/0.54**	**9**	0.54/0.43	405
	Aria	0.43/0.39	0.48/0.46	**0.6/0.57**	**21**	0.51/0.42	993
SF	LVIS	0.45/0.36	0.50/0.41	**0.63/0.53**	**7**	0.49/0.43	368
	ADE	0.4/0.33	0.45/0.40	**0.56/0.53**	**5**	0.46/0.42	295
	Aria	0.38/0.35	0.44/0.41	**0.54/0.52**	**14**	0.42/0.40	627
DL	LVIS	0.46/0.39	0.52/0.46	**0.63/0.55**	**9**	0.51/0.43	405
	ADE	0.41/0.35	0.45/0.40	**0.57/0.53**	**7**	0.46/0.41	322
	Aria	0.41/0.36	0.46/0.42	**0.56/0.54**	**18**	0.43/0.41	847

Table 2. Evaluation results on segmentation. The two numbers before and after "/" show the b-IOU and c-IOU. The first GFLOPs column shows the computational cost for AD, LTD, SOLO, and the second GFLOPs column shows that of FR.

.

outperforms the other methods in terms of both b-IOU and c-IOU. Specifically, SOLONet with HRNet as the backbone (HR) achieves the best overall results, as HRNet is larger and more powerful compared to SegFormer and DeepLabV3. In comparison, AD achieves the worst performance because average downsampling evenly downgrades the size of the all the objects within the input and eliminates key information within IOI, significantly degrading the accuracy.

5.2 Accuracy Comparison with Other Networks

In this section, we compare SOLONet with other state-of-the-art segmentation methods including Mask2Former [20] and OneFormer [43]. Both Mask2Former and OneFormer are unified frameworks for instance segmentation that add task specific modules on top of ViT-based backbone models including Swin-Large [64], Swin-Base, Swin-Small, Swin-Tiny, and Dinat-Large [35], this leads to multiple baseline algorithms including Mask2Former-Swin-Large (M2F-S-L), Mask2Former-Swin-Base (M2F-S-B), Mask2Former-Swin-Small (M2F-S-S), Mask2Former-Swin-Tiny (M2F-S-T), OneFormer-Swin-Large (OF-S-L), OneFormer-Dinat-Large (OF-D-L). All algorithms are trained and evaluated on the LVIS dataset. We averagely downsample the images to 60×60 for the Mask2Former and OneFormer models to control the computational cost in FLOPs to be roughly the same as SOLONet.

The results, shown in Figure 12 (a), reveal that SOLONet consistently achieves a significantly higher c-IOU. This highlights the effectiveness of SOLONet's saliency mechanism, which preserves a high input resolution for the IOI while compressing less crucial regions.

5.3 SOLO Streaming Algorithm Evaluation

The SOLO Streaming Algorithm described in Section 3.5 skips some segmentation computations by reusing the results from previous frames. However, this may impact the average segmentation accuracy across frames. To quantify this impact, we assess the performance of HR with SSA applied on the Aria Everyday dataset across different α and β settings, where α and β are the thresholds used by the

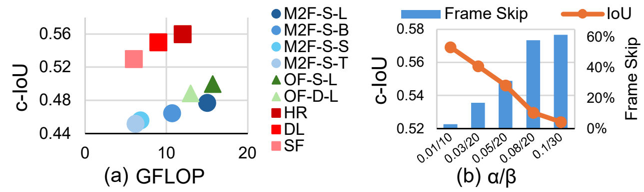

Figure 12. (a) SOLONet evaluation under same computational budget, with results for LVIS. (b) Reuse accuracy. X-axis represents the values of α and β before and after "/".

SSA in Section 3.5 for result reuse. We evaluate the average c-IoU across all frames of the Aria Everyday video, using the user gaze trace associated with each frame. The SSA is then executed to output the IOI mask for each frame. We modify the settings for α and β to study the tradeoff between segmentation accuracy and computational efficiency (Figure 12 (b)). As α and β increase, a greater number of frames are skipped by reusing results from previous frames. This results in only a slight decrease in c-IoU. For instance, when 60% of the frames are skipped, the c-IoU decreases by 0.05.

5.4 Accuracy Impact of Downsample Rate

We compare the b-IoU and c-IoU of HR trained with different size of the downsampled image I_f^s on LVIS and Aria Everyday datasets. For LVIS, we compare the results when the images are downsampled to the size of 120×120, 60×60, 40×40. For Aria Everyday, the images are downsampled to the size of 150×150, 90×90, and 60×60. The results are shown in Figure 13 (a). It can be observed that a smaller I_f^s results in lower b-IOU and c-IOU for SOLO, indicating that increasing the downsampled image size can improve accuracy.

6 Hardware Performance Evaluation

In this section, we evaluate the SOLO hardware performance, with additional results provided in supplementary materials.

6.1 Evaluation Setting

The proposed SOLO accelerator described in Section 4.2 is implemented in Verilog, with RTL synthesized using Synopsys Design Compiler [12] to estimate chip area, timing, and power, based on 45nm CMOS technology [1]. The SOLO accelerator operates at 1 GHz, with on-chip buffers modeled using CACTI [11]. Synthesis results were scaled to 22nm using DeepScaleTool [95] for alignment with current ARVR technology. The total area is $4.7mm^2$, where on-chip buffers dominate (69%), followed by the computational engine (24%), input pre-processor (6%), and sensor controller (1%). We evaluate the GPU workload latency and energy consumption on the NVIDIA Jetson Orin NX edge GPU [24] using the NVIDIA Tegra System Profiler [80].

For the saliency-based sensor simulation, we adopt the CMOS architecture developed in [59, 96], with a pixel array

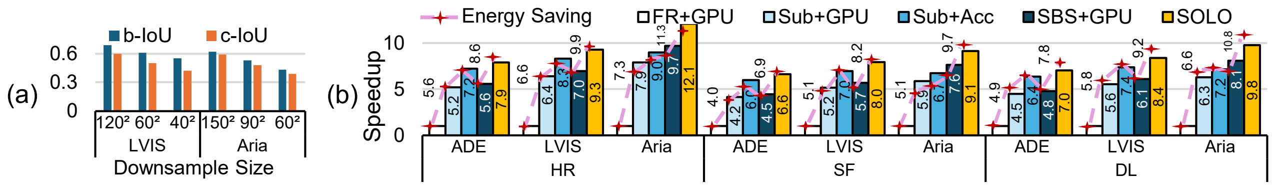

Figure 13. (a) SOLO IoU across different downsampled image sizes. (b) SOLO hardware performance evaluation.

of size 1440 × 1440. Each Pixel Sub-array (PS) includes 2 × 2 photodiodes. The ADC is arranged in a 2D array of size 8 × 360 = 2880, where each ADC handles a sub-group of 180 interleaved PSs, as described in Section 4.1. The pixel array is simulated using a standard CMOS 65 nm process node, while the remaining part of the image sensor utilizes a 22 nm process node. To support heterogeneous integration, we adopt a stacked design and model the TSV latency as 0.134 ns/access and energy as 3.492 fJ/bit, following [60, 69, 101]. In line with representative AR display designs, we model the latency of AR display as 2 ms [128, 137] and the power as 50 mW [91, 140]. The exposure time is set to 5 ms under normal lighting condition [56, 71]. We evaluate the system performance of the SOLONet using HRNet, DeepLab, and Segformer as segmentation backbones, denoted as HR, DL, and SF, in executing the instance segmentation tasks on the LVIS, ADE20K, and Aria Everyday datasets with identical algorithmic settings in Section 5. The values of α and β in the SSA (Section 3.5) are set to 0, ensuring no frame skip.

Latency (ms)	HR ADE	HR LVIS	HR Aria	SF ADE	SF LVIS	SF Aria	DL ADE	DL LVIS	DL Aria
FR+GPU	326.6	430.1	598.2	237.0	306.8	423.3	262.5	342.1	473.3
SOLO	**41.4**	**46.3**	**49.4**	**35.8**	**38.6**	**46.3**	**37.4**	**40.8**	**48.4**

Table 3. Latency numbers of FR+GPU and SOLO.

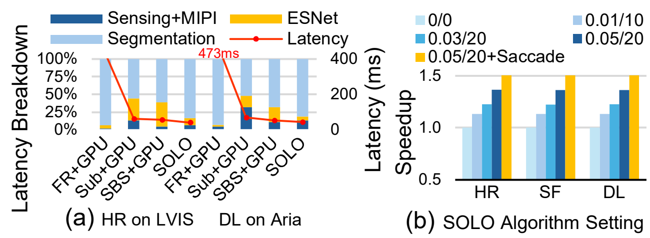

Figure 14. (a) Latency breakdowns for different baselines. The display latency is omitted given its small cost. (b) Speedup across different SSA settings. The numbers before and after "/" show the settings for α and β.

6.2 SOLO Hardware Evaluation Results

SOLO is a joint optimization framework that includes innovative algorithmic design (SOLO algorithm) and novel hardware design (SOLO accelerator and SBS). To evaluate the individual contributions of each part to the overall system performance, we use several baseline approaches. The first baseline, called *FR+GPU*, follows traditional methods by transmitting the full resolution image from the conventional sensor to the AR SoC and executing the GT-ViT and conventional segmentation DNN entirely on the GPU. The second baseline, *Sub+GPU*, transmits the full resolution image from the conventional sensor to DRAM and runs the entire SOLONet and SBS on the GPU to segment the IOI within the image. In the third baseline, *Sub+Acc*, we use the SOLO accelerator to execute the ESNet within SOLONet, while keeping the other settings the same as Sub+GPU. The fourth baseline (SBS+GPU) uses a saliency-based sensor for SBS while keeping the entire SOLONet running on the GPU. Lastly, SOLO enhances SBS+GPU by running ESNet on the SOLO accelerator and the remaining parts on the GPU. The conventional sensor adopts a rolling-shutter design with a column-parallel ADC architecture consisting of a 4 × 720 = 2880 ADC array, identical to our saliency-based sensor and aligned with prior designs [94, 96, 103]. Comparing FR+GPU with Sub+GPU highlights the improvements brought by the SOLO algorithm. Comparing Sub+Acc with Sub+GPU demonstrates the benefit of the SOLO accelerator in accelerating the ESNet. Lastly, comparing SBS+GPU with Sub+GPU reveals the contribution of SBS to the overall system performance.

Figure 13 (b) compares the above methods in terms of energy savings and execution speedups on multiple segmentation models and datasets. On average, SOLO achieves an 8.6× speedup and 9.1× energy saving compared to FR+GPU across multiple segmentation neural network architectures and datasets, while achieving better segmentation accuracy on IOI as shown in Table 2. The superior performance of SOLO is attributed to algorithmic improvements to segment image at a much smaller size, along with the specialized hardware design of the AR SoC featuring the saliency-based sensing and SOLO accelerator.

Figure 14 (a) presents the latency breakdown for various baselines running the HR model on LVIS and the DL model on Aria Everyday dataset. We can observe that SOLO outperforms the FR+GPU by enabling a much smaller hardware cost on image segmentation and sensing+MIPI communication, thanks to the SBS and the associated reduction on input size of the segmentation. Additionally, SOLO outperforms Sub+GPU by significantly reducing MIPI communication latency, and achieves better performance than SBS+GPU

Latency (ms)	HR ADE	HR LVIS	HR Aria	SF ADE	SF LVIS	SF Aria	DL ADE	DL LVIS	DL Aria
Sub+GPU	62.6	67.0	75.6	56.9	59.3	71.9	58.6	61.6	75.1
Sub+NPU	54.1	59.2	70.4	48.4	51.5	66.7	50.1	53.8	69.9
Sub+Acc	45.2	51.8	66.6	39.6	44.1	62.9	41.2	46.3	66.0
SBS+GPU	58.7	61.5	61.8	53.1	53.8	55.4	54.7	56.0	58.4
SBS+NPU	50.2	53.7	55.6	44.6	46.0	52.1	46.2	48.2	54.2
SOLO	**41.4**	**46.3**	**49.4**	**35.8**	**38.6**	**46.3**	**37.4**	**40.8**	**48.4**

Table 4. Latency comparison between NPU, GPU, and SOLO.

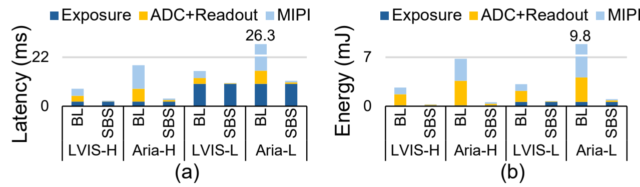

Figure 15. (a) Latency and (b) energy evaluation of saliency-based sensor. BL denotes the conventional image sensor. H and L denote the high and low light intensity conditions.

through faster ESNet processing. Detailed latency results are presented in Table 3 and Table 4, showing that SOLO enables the shortest end-to-end system latency among all the baseline solutions.

6.3 Impact of SOLO Streaming Algorithm

In Section 6.2, we assess SOLO performance without result reuse. This section further explores enhancements from the result reuse in the SSA (Section 3.5). Specifically, we adjust the settings for α and β. A larger value for α and β will enhance the average hardware performance, as it leads to more frames being skipped and more results being reused. However, this improvement comes at the cost of accuracy, as shown in Section 5.3. Additionally, we aim to evaluate the system performance improvements from saccades.

The results are shown in Figure 14 (b). We observe that changing $\alpha = 0$ and $\beta = 0$ to $\alpha = 0.05$ and $\beta = 20$ causes an 1.35× speedup on average. Moreover, as shown in Figure 12 (b), this setting still achieves good segmentation performance. Finally, incorporating additional result reuse due to saccades further enhances performance by 1.1×, leading to a 1.49× speedup compared to the baseline without result reuse.

6.4 Comparison with Neural Processing Unit

Some of the recent AR devices also include a Neural Processing Unit (NPU) for improved execution of AI tasks. For example, the Meta Quest Pro is equipped with the Qualcomm XR2 NPU [88]. To compare SOLO with an existing NPU, we develop two additional baselines. The first baseline, Sub+NPU, executes ESNet on the Qualcomm XR2 NPU, with all other settings identical to those of the Sub+Acc baseline described in Section 6.2. The second baseline, SBS+NPU, executes ESNet on the Qualcomm XR2 NPU, with other settings identical to those of SOLO. Table 4 summarizes the latency comparison. NPU latencies are measured on the Qualcomm XR2 mobile platform [88] using the Qualcomm AI Hub toolkit [87]. We observe that Sub+NPU and SBS+NPU achieve lower latency than Sub+GPU and SBS+GPU, yet both remain inferior to SOLO, underscoring the necessity of SOLO for AR implementations.

6.5 Saliency Based Sensing Performance Analysis

6.5.1 Impact of Saliency Based Sensing on Frame Latency. To quantify the impact of SBS on frame latency, we compare the *Sub+GPU* and *SBS+GPU* baselines shown in Figure 13 (b). On average, SBS+GPU reduces the frame latency by 1.2× over Sub+GPU, with more pronounced benefits for high-resolution images. The latency reduction achieved by SBS stems from shortening both the ADC+readout, and MIPI transfer stages, which are otherwise time-consuming.

6.5.2 Latency and Energy Impact of SBS on Image Sensing and MIPI transfer. Figure 15 presents detailed latency and energy results for the sensor, separated into exposure, ADC+readout, and MIPI transfer stages. We compare the sensing cost of the saliency-based sensor in SOLO with that of a conventional column-parallel sensor, using the hardware settings described in Section 6.2. Since exposure time depends on lighting conditions, we evaluate across high light intensity and low light intensity conditions, where the exposure time is set to 2 ms and 10 ms [56, 71], respectively. On average, saliency-based sensor achieves a 4.3× reduction in latency in high light intensity conditions. For example, when processing 960 × 960 images in Aria, after a fixed 2 ms exposure time in high light intensity condition, the conventional image sensor in the Sub+GPU scheme converts and transfers all pixel values, incurring 5.8 ms ADC+readout latency and 10.5 ms MIPI transfer latency. In contrast, the SBS scheme reduces the number of captured pixels and sensing rounds, lowering ADC+readout latency to 0.7 ms and MIPI transfer latency to 0.6 ms. In low-light conditions, where exposure time dominates, the latency improvement is less pronounced, yet SBS still achieves an average 1.9× reduction compared to conventional sensor. For energy, where ADC+readout and MIPI transfer are the primary bottlenecks, SBS provides 8.9× saving on average.

6.6 User Study

To assess the improvement in user experience provided by SOLO over the conventional FR+GPU approach (Figure 13), we simulate their visual effects on the Meta Quest Pro HMD [86]. For each test image, segmentation masks for the IOIs were precomputed using HR and Mask2Former (M2F-S-L) [20], following the settings in Section 5.2. During the user study,

Figure 16. Participants are joining in the user study, which consists of 32 trials.

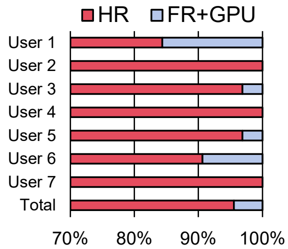

Figure 17. Selection results from the participants.

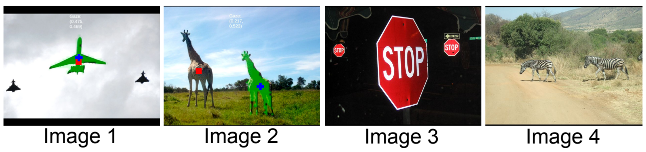

Figure 18. Four screenshots captured during the user study. In Image 1 and 2, red dots indicate current gaze locations, blue cross represents the gaze location used for segmentation, reflecting a prior state due to latency.

the HMD displays the mask of the currently observed IOI according to gaze locations obtained by the gaze tracker, simulating the visual effects of both algorithms.

To account for system performance, we simulate processing latencies using the settings in Section 6.1, and artificially introduce the delay between the moment the user's gaze identifies the IOI and when the segmentation mask is displayed on the VR screen. Due to its reduced input size, HR exhibited significantly lower average latency (42.6 ms) compared to M2F-S-L (547 ms). Figure 18 illustrates these differences: in Image 1, HR maintains low latency, ensuring masks closely align with current gaze location, whereas in Image 2, FR+GPU's higher latency results in noticeable misalignment between the mask and current gaze location.

Seven participants take part in the study (in Figure 16, where the computer monitor display the HMD-cast content), and interact using the HMD controllers. The stimuli consist of four images (Figure 18). The two methods, denoted as m_1 (HR) and m_2 (FR+GPU), are directly compared. Participants perform a two-interval forced-choice (2IFC) task [124], viewing two segmentation results (t_1 and t_2) per image, with masks and latency applied. Each participant completes 32 trials of 4 images, each tested with both t and m pairs across 4 repetitions in random order. Figure 17 presents the results that HR is preferred in 96%±6% of trials, consistently outperforming the baseline across individual images (96%±6% for image 1, 91%±16% for image 2, 100%±0% for image 3, and 95%±10% for image 4). These results evidence that SOLO can improve AR user experience.

To evaluate SOLO's robustness on dynamic scenes, we test it on the DAVIS 2016 dataset [85], which contains videos with

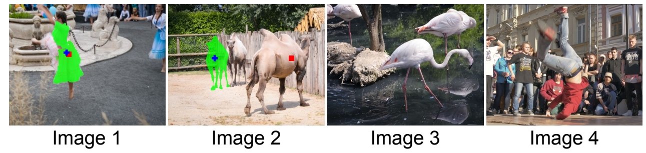

Figure 19. Four screenshots of user study on DAVIS 2016.

moving targets and dynamic backgrounds. We randomly select one point in the video frame as the gaze location and train the SOLO with HR network, which achieves b-IoU and c-IoU of 0.56 and 0.49 on DAVIS 2016 dataset. The frame size is set to 480×480 in our experiment and HR downsamples the frames to the size of 60×60. In contrast, we evaluate the original M2F-S-L to segment the full resolution image and select the IOI mask, resulting in the b-IoU and c-IoU of 0.44 and 0.41, showing the superiority of SOLO on this dataset.

Additionally, we select four representative videos from the validation dataset and conduct the user study, with one frame from each of the videos shown in Figure 19. To simulate processing latency, we introduce a delay of 33 ms for HR and 478 ms for M2F-S-L on the 480×480 resolution frames. We conduct an in-depth A/B preference test with four participants (32 trials per participant, 128 trials total). Under the null hypothesis that SOLO and the baseline are equally likely to be chosen ($P = 0.5$), a one-sided binomial test [74] showed that SOLO was selected in 122 of 128 trials, yielding a highly significant result ($P < 1.67 \times 10^{-29}$). This demonstrates a strong preference for SOLO in dynamic scenes.

We also evaluate HR with SSA applied on the DAVIS 2016 dataset. Although in DAVIS 2016 dataset, the scene is more dynamic compared to Aria Everyday dataset, SSA still skips 13% of frames. The c-IoU changes from 0.49 to 0.48, demonstrating SSA's robustness in accurately determining skip conditions. Despite the reduced reuse opportunities, SOLO maintains an average end-to-end per-frame latency of 28.7 ms with SSA applied, well within the 50 ms real-time budget.

7 Conclusion

Image segmentation is crucial for AR applications, but processing full resolution input frames is significantly costly. SOLO tackles these challenges by utilizing the natural dynamics of human eye behavior to minimize the computational load of the segmentation process.

The saliency-driven subsampling principle can also extend to other vision tasks that rely on user attention. Overall, SOLO not only enables efficient AI applications on AR devices but also opens up a new field of research in integrating human attention with AI system design.

Acknowledgement

We acknowledge Meta Reality Labs for their valuable technical discussions and support of this work.

References

[1] [n. d.]. Nangate freepdk45 open cell library. https://silvaco.com/services/library-design/

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

[3] [n. d.]. Vive Pro 2. https://www.vive.com/us/product/vive-pro2/specs/.

[4] [n. d.]. What is Mobile Industry Processor Interface (MIPI) Protocol? https://www.synopsys.com/blogs/chip-design/what-is-mobile-industry-processor-interface-protocol.html

[5] Abdullah M Al-Ansi, Mohammed Jaboob, Askar Garad, and Ahmed Al-Ansi. 2023. Analyzing augmented reality (AR) and virtual reality (VR) recent development in education. *Social Sciences & Humanities Open* 8, 1 (2023), 100532.

[6] Rachel Albert, Anjul Patney, David Luebke, and Joohwan Kim. 2017. Latency requirements for foveated rendering in virtual reality. *ACM Transactions on Applied Perception (TAP)* 14, 4 (2017), 1–13.

[7] Hassan Abu Alhaija, Siva Karthik Mustikovela, Lars Mescheder, Andreas Geiger, and Carsten Rother. 2017. Augmented reality meets deep learning for car instance segmentation in urban scenes. In *British machine vision conference*, Vol. 1.

[8] Toshiki Arai, Toshio Yasue, Kazuya Kitamura, Hiroshi Shimamoto, Tomohiko Kosugi, Sung-Wook Jun, Satoshi Aoyama, Ming-Chieh Hsu, Yuichiro Yamashita, Hirofumi Sumi, et al. 2017. A 1.1-33-Mpixel 240-fps 3-D-Stacked CMOS Image Sensor With Three-Stage Cyclic-Cyclic-SAR Analog-to-Digital Converters. *IEEE Transactions on Electron Devices* 64, 12 (2017), 4992–5000.

[9] Rishi Athavale, Lakshmi Sritan Motati, and Rohan Kalahasty. 2022. One Eye is All You Need: Lightweight Ensembles for Gaze Estimation with Single Encoders. arXiv:2211.11936 [cs.CV] https://arxiv.org/abs/2211.11936

[10] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. 2015. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39 (2015), 2481–2495. https://api.semanticscholar.org/CorpusID:60814714

[11] Rajeev Balasubramonian, Andrew B. Kahng, Naveen Muralimanohar, Ali Shafiee, and Vaishnav Srinivas. 2017. CACTI 7: New Tools for Interconnect Exploration in Innovative Off-Chip Memories. *ACM Trans. Archit. Code Optim.* 14, 2, Article 14 (June 2017), 25 pages. doi:10.1145/3085572

[12] B. Jayant Baliga. 2019. Synopsys. *Wide Bandgap Semiconductor Power Devices* (2019). https://api.semanticscholar.org/CorpusID:239327327

[13] Robert W Baloh, Andrew W Sills, Warren E Kumley, and Vicente Honrubia. 1975. Quantitative measurement of saccade amplitude, duration, and velocity. *Neurology* 25, 11 (1975), 1065–1065.

[14] Cyrus S Bamji, Swati Mehta, Barry Thompson, Tamer Elkhatib, Stefan Wurster, Onur Akkaya, Andrew Payne, John Godbaz, Mike Fenton, Vijay Rajasekaran, et al. 2018. IMpixel 65nm BSI 320MHz demodulated TOF Image sensor with 3μm global shutter pixels and analog binning. In *2018 IEEE International Solid-State Circuits Conference-(ISSCC)*. IEEE, 94–96.

[15] Eleonora Bottani and Giuseppe Vignali. 2019. Augmented reality technology in the manufacturing industry: A review of the last decade. *Iise Transactions* 51, 3 (2019), 284–310.

[16] Bert De Brabandere, Davy Neven, and Luc Van Gool. 2017. Semantic Instance Segmentation with a Discriminative Loss Function. *ArXiv* abs/1708.02551 (2017). https://api.semanticscholar.org/CorpusID:2328623

[17] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. 2017. Rethinking Atrous Convolution for Semantic Image Segmentation. arXiv:1706.05587 [cs.CV] https://arxiv.org/abs/1706.05587

[18] Song Chen, Chiao Liu, Lyle Bainbridge, Qing Chao, Ramakrishna Chilukuri, Wei Gao, Andrew P Hammond, Tsung-Hsun Tsai, Ken Miyauchi, Isao Takayanagi, et al. 2023. A 3.96 μm, 124dB Dynamic Range, 6.2 mW Stacked Digital Pixel Sensor with Monochrome and Near-Infrared Dual-Channel Global Shutter Capture. In *2023 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*. IEEE, 1–2.

[19] Bowen Cheng, Maxwell D. Collins, Yukun Zhu, Ting Liu, Thomas S. Huang, Hartwig Adam, and Liang-Chieh Chen. 2019. Panoptic-DeepLab: A Simple, Strong, and Fast Baseline for Bottom-Up Panoptic Segmentation. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), 12472–12482. https://api.semanticscholar.org/CorpusID:208248153

[20] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. 2022. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 1290–1299.

[21] Bowen Cheng, Alexander G. Schwing, and Alexander Kirillov. 2021. Per-Pixel Classification is Not All You Need for Semantic Segmentation. In *Neural Information Processing Systems*. https://api.semanticscholar.org/CorpusID:235829267

[22] Jaehyuk Choi, Seokjun Park, Jihyun Cho, and Euisik Yoon. 2015. An energy/illumination-adaptive CMOS image sensor with reconfigurable modes of operations. *IEEE Journal of Solid-State Circuits* 50, 6 (2015), 1438–1450.

[23] Jaehyuk Choi, Jungsoon Shin, Dongwu Kang, and Du-Sik Park. 2015. Always-on CMOS image sensor for mobile and wearable devices. *IEEE Journal of Solid-State Circuits* 51, 1 (2015), 130–140.

[24] NVIDIA Corporation. [n. d.]. Jetson AGX Orin. Online. https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin/ Accessed: 2024-11-07.

[25] Mark R Diamond, John Ross, and Maria C Morrone. 2000. Extraretinal control of saccadic suppression. *Journal of Neuroscience* 20, 9 (2000), 3449–3455.

[26] Alexey Dosovitskiy. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[27] Jakob Engel, Kiran Somasundaram, Michael Goesele, Albert Sun, Alexander Gamino, Andrew Turner, Arjang Talattof, Arnie Yuan, Bilal Souti, Brighid Meredith, et al. 2023. Project aria: A new tool for egocentric multi-modal ai research. *arXiv preprint arXiv:2308.13561* (2023).

[28] Yu Feng, Nathan Goulding-Hotta, Asif Khan, Hans Reyserhove, and Yuhao Zhu. 2022. Real-Time Gaze Tracking with Event-Driven Eye Segmentation. In *2022 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. 399–408. doi:10.1109/VR51125.2022.00059

[29] Yu Feng, Tianrui Ma, Yuhao Zhu, and Xuan Zhang. 2024. BlissCam: Boosting Eye Tracking Efficiency with Learned In-Sensor Sparse Sampling. *arXiv preprint arXiv:2404.15733* (2024).

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

[31] Santiago González Izard, Juan A Juanes Méndez, Pablo Ruisoto Palomera, and Francisco J García-Peñalvo. 2019. Applications of virtual and augmented reality in biomedical imaging. *Journal of medical systems* 43, 4 (2019), 102.

[32] Santiago González Izard, Ramiro Sánchez Torres, Oscar Alonso Plaza, Juan Antonio Juanes Mendez, and Francisco José García-Peñalvo. 2020. Nextmed: automatic imaging segmentation, 3D reconstruction, and 3D model visualization platform using augmented and virtual reality. *Sensors* 20, 10 (2020), 2962.

[33] Agrim Gupta, Piotr Dollár, and Ross Girshick. 2019. LVIS: A Dataset for Large Vocabulary Instance Segmentation. arXiv:1908.03195 [cs.CV] https://arxiv.org/abs/1908.03195

[34] Dan Witzner Hansen and Qiang Ji. 2010. In the Eye of the Beholder: A Survey of Models for Eyes and Gaze. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32, 3 (2010), 478–500. doi:10.1109/TPAMI.2009.30

[35] Ali Hassani and Humphrey Shi. 2022. Dilated neighborhood attention transformer. *arXiv preprint arXiv:2209.15001* (2022).

[36] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. 2017. Mask R-CNN. https://api.semanticscholar.org/CorpusID:54465873

[37] Tairan He, Zhengyi Luo, Xialin He, Wenli Xiao, Chong Zhang, Weinan Zhang, Kris Kitani, Changliu Liu, and Guanya Shi. 2024. OmniH2O: Universal and Dexterous Human-to-Humanoid Whole-Body Teleoperation and Learning. *arXiv preprint arXiv:2406.08858* (2024).

[38] Min-Chai Hsieh and Yu-Hsuan Lin. 2017. VR and AR applications in medical practice and education. *Hu Li Za Zhi* 64, 6 (2017), 12–18.

[39] https://www.imveurope.com/. 2022. 3D stacked technologies applied to custom CMOS image sensors – should you invest in differentiation? https://www.laserfocusworld.com/detectors-imaging/article/55248139/sony-researchers-combine-color-imaging-and-ranging-in-one-sensor

[40] Jang-Su Hyeon, Sang-Hyeon Kim, and Hyeon-June Kim. 2022. A low-power CMOS image sensor with multiple-column-parallel readout structure. *IEEE Journal of the Electron Devices Society* 10 (2022), 180–187.

[41] Rimon Ikeno, Kazuya Mori, Masayuki Uno, Ken Miyauchi, Toshiyuki Isozaki, Isao Takayanagi, Junichi Nakamura, Shou-Gwo Wuu, Lyle Bainbridge, Andrew Berkovich, et al. 2022. A 4.6-μm, 127-dB dynamic range, ultra-low power stacked digital pixel sensor with overlapped triple quantization. *IEEE Transactions on Electron Devices* 69, 6 (2022), 2943–2950.

[42] Meta Platform Inc. 2022. Meta Quest Pro. https://www.meta.com/quest/quest-pro/.

[43] Jitesh Jain, Jiachen Li, Mang Tik Chiu, Ali Hassani, Nikita Orlov, and Humphrey Shi. 2023. Oneformer: One transformer to rule universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.* 2989–2998.

[44] Gouri Jha, Lavanya shm Sharma, and Shailja Gupta. 2021. Future of augmented reality in healthcare department. In *Proceedings of Second International Conference on Computing, Communications, and Cyber-Security: IC4S 2020.* Springer, 667–678.

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

[46] Jaehoon Jun. 2023. A comprehensive methodology for optimizing read-out timing and reference DAC offset in high frame rate image sensing systems. *Sensors* 23, 16 (2023), 7048.

[47] Jaehoon Jun, Haneol Seo, Hyukbin Kwon, Jongyeon Lee, Beomsoo Yoon, Youngwoo Lee, Yongbin Kim, Woong Joo, Jesuk Lee, and Kyoungmin Koh. 2022. A 0.7 μm-pitch 108 Mpixel nonacell-based CMOS image sensor with decision-feedback technique. In *2022 IEEE International Symposium on Circuits and Systems (ISCAS).* IEEE, 283–287.

[48] Tae-young Ko and Seung-ho Lee. 2020. Novel method of semantic segmentation applicable to augmented reality. *Sensors* 20, 6 (2020), 1737.

[49] Bart Krekelberg. 2010. Saccadic suppression. *Current Biology* 20, 5 (2010), R228–R229.

[50] Wonseok Lee, Kyeongjong Lim, Jeonghyeon Cheon, Soyi Jeong, Jinyeon Lim, Youngsung Cho, Shusaku Ishikawa, Seongwon Jo, Seongwook Song, Minsu Kang, Kyungil Kim, Seunghyun Lim, Youngjin Kim, Sunghoo Choi, and Jungchan Kyoung. 2023. A Multi-Pixel Compression for Low-Power Imaging System and Architecture. In *2023 IEEE International Symposium on Circuits and Systems (ISCAS).* 1–5. doi:10.1109/ISCAS46773.2023.10181603

[51] Walter D Leon-Salas, Sina Balkir, Khalid Sayood, Nathan Schemm, and Michael W Hoffman. 2007. A CMOS imager with focal plane compression using predictive coding. *IEEE Journal of Solid-State Circuits* 42, 11 (2007), 2555–2572.

[52] Bin Li, Hong Fu, Desheng Wen, and WaiLun LO. 2018. Etracker: A Mobile Gaze-Tracking System with Near-Eye Display Based on a Combined Gaze-Tracking Algorithm. *Sensors* 18, 5 (2018). doi:10.3390/s18051626

[53] Hao Li, Hui Zhang, Xiaolian Guo, and Guangshu Hu. 2009. Image restoration after pixel binning in image sensors. *Tsinghua Science and Technology* 14, 4 (2009), 541–545.

[54] Xiangtai Li, Henghui Ding, Wenwei Zhang, Haobo Yuan, Jiangmiao Pang, Guangliang Cheng, Kai Chen, Ziwei Liu, and Chen Change Loy. 2023. Transformer-Based Visual Segmentation: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46 (2023), 10138–10163. https://api.semanticscholar.org/CorpusID:258212528

[55] Yanwei Li, Hengshuang Zhao, Xiaojuan Qi, Liwei Wang, Zeming Li, Jian Sun, and Jiaya Jia. 2020. Fully Convolutional Networks for Panoptic Segmentation. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020), 214–223. https://api.semanticscholar.org/CorpusID:227239035

[56] Hao Lin, Darragh Mullins, Dara Molloy, Enda Ward, Fiachra Collins, Patrick Denny, Martin Glavin, Brian Deegan, and Edward Jones. 2024. Optimizing Camera Exposure Time for Automotive Applications. *Sensors* 24, 16 (2024), 5135.

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

[58] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision.* 2980–2988.

[59] Chiao Liu, Lyle Bainbridge, Andrew Berkovich, Song Chen, Wei Gao, Tsung-Hsun Tsai, Kazuya Mori, Rimon Ikeno, Masayuki Uno, Toshiyuki Isozaki, Yu-Lin Tsai, Isao Takayanagi, and Junichi Nakamura. 2020. A 4.6m, 512×512, Ultra-Low Power Stacked Digital Pixel Sensor with Triple Quantization and 127dB Dynamic Range. In *2020 IEEE International Electron Devices Meeting (IEDM).* 16.1.1–16.1.4. doi:10.1109/IEDM13553.2020.9371913

[60] Chiao Liu, Andrew Berkovich, Song Chen, Hans Reyserhove, Syed Shakib Sarwar, and Tsung-Hsun Tsai. 2019. Intelligent vision systems–bringing human-machine interface to AR/VR. In *2019 IEEE International Electron Devices Meeting (IEDM).* IEEE, 10–5.

[61] Chiao Liu, Song Chen, Tsung-Hsun Tsai, Barbara De Salvo, and Jorge Gomez. 2022. Augmented reality-the next frontier of image sensors and compute systems. In *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, Vol. 65. IEEE, 426–428.

[62] Chiao Liu, Michael Hall, Renzo De Nardi, Nicholas Trail, and Richard Newcombe. 2017. Sensors for future VR applications. In *2017 International Image Sensor Workshop (IISW).* 250–253.

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

[64] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision.* 10012–10022.

[65] Conny Lu, Praneeth Chakravarthula, Kaihao Liu, Xixiang Liu, Siyuan Li, and Henry Fuchs. 2022. Neural 3D Gaze: 3D Pupil Localization and Gaze Tracking based on Anatomical Eye Model and Neural Refraction Correction. In *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR).* 375–383. doi:10.1109/ISMAR55827.2022.00053

[66] Feng Lu, Takahiro Okabe, Yusuke Sugano, and Yoichi Sato. 2011. A Head Pose-free Approach for Appearance-based Gaze Estimation. In *British Machine Vision Conference*. https://api.semanticscholar.org/CorpusID:7733236

[67] 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. 2024. Aria Everyday Activities Dataset. arXiv:2402.13349 [cs.CV] https://arxiv.org/abs/2402.13349

[68] Longfei Ma, Zhencheng Fan, Guochen Ning, Xinran Zhang, and Hongen Liao. 2018. 3D visualization and augmented reality for orthopedics. *Intelligent Orthopaedics: Artificial Intelligence and Smart Image-guided Technology for Orthopaedics* (2018), 193–205.

[69] Tianrui Ma, Yu Feng, Xuan Zhang, and Yuhao Zhu. 2023. Camj: Enabling system-level energy modeling and architectural exploration for in-sensor visual computing. In *Proceedings of the 50th Annual International Symposium on Computer Architecture*. 1–14.

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

[71] Mitsuhito Mase, Shoji Kawahito, Masaaki Sasaki, Yasuo Wakamori, and Masanori Furuta. 2005. A wide dynamic range CMOS image sensor with multiple exposure-time signal outputs and 12-bit column-parallel cyclic A/D converters. *IEEE journal of solid-state circuits* 40, 12 (2005), 2787–2795.

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

[73] Pier Luigi Mazzeo, Dilan D'Amico, Paolo Spagnolo, and Cosimo Distante. 2021. Deep Learning based Eye gaze estimation and prediction. In *2021 6th International Conference on Smart and Sustainable Technologies (SpliTech)*. 1–6. doi:10.23919/SpliTech52315.2021.9566413

[74] John H McDonald. 2014. Handbook of biological statistics. (2014).

[75] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. 2021. Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence* 44, 7 (2021), 3523–3542.

[76] Shervin Minaee, Yuri Boykov, Fatih Murat Porikli, Antonio J. Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. 2020. Image Segmentation Using Deep Learning: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44 (2020), 3523–3542. https://api.semanticscholar.org/CorpusID:210702798

[77] Concetta Morrone, David Burr, et al. 2009. Visual stability during saccadic eye movements. *The cognitive neurosciences, Ed* 4 (2009), 511–524.

[78] Andrew YC Nee, Soh Khim Ong, George Chryssolouris, and Dimitris Mourtzis. 2012. Augmented reality applications in design and manufacturing. *CIRP annals* 61, 2 (2012), 657–679.

[79] Davy Neven, Bert De Brabandere, Marc Proesmans, and Luc Van Gool. 2019. Instance Segmentation by Jointly Optimizing Spatial Embeddings and Clustering Bandwidth. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), 8829–8837. https://api.semanticscholar.org/CorpusID:195658142

[80] NVIDIA Corporation. 2025. *Tegra System Profiler*. NVIDIA GameWorks Documentation. Retrieved 2025-08-09 from https://docs.nvidia.com/gameworks/content/developertools/mobile/tegra_system_profiler/tegraprofiler_main.htm Available documentation: 3.7 / 3.6 / 3.1 / 2.8 / 2.7 / 2.5 User Guides.

[81] Yusuke Oike. 2021. Evolution of image sensor architectures with stacked device technologies. *IEEE Transactions on Electron Devices* 69, 6 (2021), 2757–2765.

[82] Cristina Palmero, Abhishek Sharma, Karsten Behrendt, Kapil Krishnakumar, Oleg V Komogortsev, and Sachin S Talathi. 2021. Openeds2020 challenge on gaze tracking for vr: Dataset and results. *Sensors* 21, 14 (2021), 4769.

[83] Junrui Pan and Timothy G Rogers. 2024. CRISP: Concurrent Rendering and Compute Simulation Platform for GPUs. In *2024 IEEE International Symposium on Workload Characterization (IISWC)*. IEEE, 1–14.

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

[85] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. 2016. A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation. In *Computer Vision and Pattern Recognition*.

[86] Meta Quest Pro. 2022. https://www.meta.com/quest/quest-pro/tech-specs/#tech-specs.

[87] Qualcomm Technologies, Inc. 2024. Qualcomm AI Hub. https://app.aihub.qualcomm.com/docs/index.html. Accessed: 2025-04-19.

[88] Qualcomm Technologies, Inc. 2024. Snapdragon XR2 Gen 2 Platform. https://www.qualcomm.com/products/mobile/snapdragon/xr-vr-ar/snapdragon-xr2-gen-2-platform. Accessed: 2025-04-19.

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

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

[91] Maria Rodriguez Fernandez, Eduardo Zalama Casanova, and Ignacio Gonzalez Alonso. 2015. Review of display technologies focusing on power consumption. *Sustainability* 7, 8 (2015), 10854–10875.

[92] Mohammadreza Saed, Yuan Hsi Chou, Lufei Liu, Tyler Nowicki, and Tor M. Aamodt. 2022. Vulkan-Sim: A GPU Architecture Simulator for Ray Tracing. In *2022 55th IEEE/ACM International Symposium on Microarchitecture (MICRO)*. 263–281. doi:10.1109/MICRO56248.2022.00027

[93] Chandan K Sahu, Crystal Young, and Rahul Rai. 2021. Artificial intelligence (AI) in augmented reality (AR)-assisted manufacturing applications: a review. *International journal of production research* 59, 16 (2021), 4903–4959.

[94] Masaki Sakakibara, Koji Ogawa, Shin Sakai, Yasuhisa Tochigi, Katsumi Honda, Hidekazu Kikuchi, Takuya Wada, Yasunobu Kamikubo, Tsukasa Miura, Masahiko Nakamizo, et al. 2018. A back-illuminated global-shutter CMOS image sensor with pixel-parallel 14b subthreshold ADC. In *2018 IEEE International Solid-State Circuits Conference-(ISSCC)*. IEEE, 80–82.

[95] Satyabrata Sarangi and Bevan Baas. 2021. DeepScaleTool: A tool for the accurate estimation of technology scaling in the deep-submicron era. In *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 1–5.

[96] Min-Woong Seo, Myunglae Chu, Hyun-Yong Jung, Suksan Kim, Jiyoun Song, Daehee Bae, Sanggwon Lee, Junan Lee, Sung-Yong Kim, Jongyeon Lee, et al. 2022. 2.45 e-RMS low-random-noise, 598.5 mW low-power, and 1.2 kfps high-speed 2-Mp global shutter CMOS image sensor with pixel-level ADC and memory. *IEEE Journal of Solid-State Circuits* 57, 4 (2022), 1125–1137.

[97] Evan Shelhamer, Jonathan Long, and Trevor Darrell. 2014. Fully convolutional networks for semantic segmentation. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2014), 3431–3440. https://api.semanticscholar.org/CorpusID:1629541

[98] Sony. 2022. 3D stacked technologies applied to custom CMOS image sensors – should you invest in differentiation? https://www.imveurope.com/article/3d-stacked-technologies-applied-

custom-cmos-image-sensors-should-you-invest-differentiation

[99] Yusuke Sugano, Yasuyuki Matsushita, and Yoichi Sato. 2014. Learning-by-Synthesis for Appearance-Based 3D Gaze Estimation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 1821–1828. doi:10.1109/CVPR.2014.235

[100] Ke Sun, Yang Zhao, Borui Jiang, Tianheng Cheng, Bin Xiao, Dong Liu, Yadong Mu, Xinggang Wang, Wenyu Liu, and Jingdong Wang. 2019. High-resolution representations for labeling pixels and regions. *arXiv preprint arXiv:1904.04514* (2019).

[101] Xiaoyu Sun, Xiaochen Peng, Sai Zhang, Jorge Gomez, Win-San Khwa, Syed Sarwar, Ziyun Li, Weidong Cao, Zhao Wang, Chiao Liu, et al. 2024. Estimating Power, Performance, and Area for On-Sensor Deployment of AR/VR Workloads Using an Analytical Framework. *ACM Transactions on Design Automation of Electronic Systems* (2024).

[102] Lech Świrski and Neil A. Dodgson. 2013. A fully-automatic, temporal approach to single camera, glint-free 3D eye model fitting [Abstract]. In *Proceedings of ECEM 2013* (Lund, Sweden). http://www.cl.cam.ac.uk/research/rainbow/projects/eyemodelfit/

[103] Tomohiro Takahashi, Yuichi Kaji, Yasunori Tsukuda, Shinichiro Futami, Katsuhiko Hanzawa, Takahito Yamauchi, Ping Wah Wong, Frederick T Brady, Phil Holden, Thomas Ayers, et al. 2018. A stacked CMOS image sensor with array-parallel ADC architecture. *IEEE Journal of Solid-State Circuits* 53, 4 (2018), 1061–1070.

[104] Khaled Takrouri, Edward Causton, and Benjamin Simpson. 2022. AR technologies in engineering education: Applications, potential, and limitations. *Digital* 2, 2 (2022), 171–190.

[105] Leonardo Tanzi, Pietro Piazzolla, Francesco Porpiglia, and Enrico Vezzetti. 2021. Real-time deep learning semantic segmentation during intra-operative surgery for 3D augmented reality assistance. *International Journal of Computer Assisted Radiology and Surgery* 16, 9 (2021), 1435–1445.

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

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

[108] Yasuhisa Tochigi, Katsuhiko Hanzawa, Yuri Kato, Rihito Kuroda, Hideki Mutoh, Ryuta Hirose, Hideki Tominaga, Kenji Takubo, Yasushi Kondo, and Shigetoshi Sugawa. 2013. A Global-Shutter CMOS Image Sensor With Readout Speed of 1-Tpixel/s Burst and 780-Mpixel/s Continuous. *IEEE Journal of Solid-State Circuits* 48, 1 (2013), 329–338. doi:10.1109/JSSC.2012.2219685

[109] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herv'e J'egou. 2020. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*. https://api.semanticscholar.org/CorpusID:229363322

[110] Tsung-Hsun Tsai, Kwuang-Han Chang, Andrew Berkovich, Raffaele Capoccia, Song Chen, Zhao Wang, Chiao Liu, Yi-Hsuan Lin, ShengYeh Lai, Hao-Ming Hsu, et al. 2025. A 400× 400 3.24 μm 117dB-Dynamic-Range 3-Layer Stacked Digital Pixel Sensor. *ITE Technical Report; ITE Tech. Rep.* 49, 13 (2025), 53–57.

[111] Rosanna Maria Viglialoro, Sara Condino, Giuseppe Turini, Marina Carbone, Vincenzo Ferrari, and Marco Gesi. 2021. Augmented reality, mixed reality, and hybrid approach in healthcare simulation: a systematic review. *Applied Sciences* 11, 5 (2021), 2338.

[112] Huiyu Wang, Yukun Zhu, Bradley Green, Hartwig Adam, Alan Loddon Yuille, and Liang-Chieh Chen. 2020. Axial-DeepLab: Stand-Alone Axial-Attention for Panoptic Segmentation. In *European Conference on Computer Vision*. https://api.semanticscholar.org/CorpusID:212737096

[113] 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. 2020. Deep High-Resolution Representation Learning for Visual Recognition. arXiv:1908.07919 [cs.CV] https://arxiv.org/abs/1908.07919

[114] Kang Wang and Qiang Ji. 2017. Real Time Eye Gaze Tracking with 3D Deformable Eye-Face Model. In *2017 IEEE International Conference on Computer Vision (ICCV)*. 1003–1011. doi:10.1109/ICCV.2017.114

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

[116] Thomas Westin, José Neves, Peter Mozelius, Carla Sousa, and Lara Mantovan. 2022. Inclusive AR-games for education of deaf children: Challenges and opportunities. In *European Conference on Games Based Learning*, Vol. 16. 597–604.

[117] Erroll Wood, Tadas Baltrušaitis, Louis-Philippe Morency, Peter Robinson, and Andreas Bulling. 2016. A 3D Morphable Eye Region Model for Gaze Estimation. In *Computer Vision – ECCV 2016*, Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (Eds.). Springer International Publishing, Cham, 297–313.

[118] Erroll Wood, Tadas Baltrušaitis, Louis-Philippe Morency, Peter Robinson, and Andreas Bulling. 2016. Learning an appearance-based gaze estimator from one million synthesised images. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications (ETRA '16)*. 131–138. doi:10.1145/2857491.2857492

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

[120] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. 2021. SegFormer: Simple and efficient design for semantic segmentation with transformers. *Advances in neural information processing systems* 34 (2021), 12077–12090.

[121] Yuwen Xiong, Renjie Liao, Hengshuang Zhao, Rui Hu, Min Bai, Ersin Yumer, and Raquel Urtasun. 2019. UPSNet: A Unified Panoptic Segmentation Network. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), 8810–8818. https://api.semanticscholar.org/CorpusID:58004609

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

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

[124] Yaffa Yeshurun, Marisa Carrasco, and Laurence T Maloney. 2008. Bias and sensitivity in two-interval forced choice procedures: Tests of the difference model. *Vision research* 48, 17 (2008), 1837–1851.

[125] Yuk-Hoi Yiu, Moustafa Aboulatta, Theresa Raiser, Leoni Ophey, Virginia L. Flanagin, Peter zu Eulenburg, and Seyed-Ahmad Ahmadi. 2019. DeepVOG: Open-source pupil segmentation and gaze estimation in neuroscience using deep learning. *Journal of Neuroscience Methods* 324 (2019), 108307. doi:10.1016/j.jneumeth.2019.05.016

[126] Michitaka Yoshida, Toshiki Sonoda, Hajime Nagahara, Kenta Endo, Yukinobu Sugiyama, and Rin-ichiro Taniguchi. 2019. High-speed imaging using CMOS image sensor with quasi pixel-wise exposure. *IEEE Transactions on Computational Imaging* 6 (2019), 463–476.

[127] Yuhao Zhu. 2022. Opportunities and Challenges of Computing in Die-Stacked Image Sensors. https://www.sigarch.org/opportunities-and-challenges-of-computing-in-die-stacked-image-sensors

[128] Roberts Zabels, Rendijs Smukulis, Ralfs Fenuks, Andris Kučiks, Elza Linina, Kriss Osmanis, and Ilmars Osmanis. 2021. Reducing motion to photon latency in multi-focal augmented reality near-eye display. In *Optical Architectures for Displays and Sensing in Augmented, Virtual, and Mixed Reality (AR, VR, MR) II*, Vol. 11765. SPIE, 213–224.

[129] Baoheng Zhang, Yizhao Gao, Jingyuan Li, and Hayden Kwok-Hay So. 2024. 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*. 5771–5779.

[130] Jiachao Zhang, Jie Jia, Andong Sheng, and Keigo Hirakawa. 2018. Pixel binning for high dynamic range color image sensor using square sampling lattice. *IEEE Transactions on Image Processing* 27, 5 (2018), 2229–2241.

[131] Milin Zhang and Amine Bermak. 2010. CMOS Image Sensor with On-Chip Image Compression: A Review and Performance Analysis. *J. Sensors* 2010 (2010), 920693:1–920693:17. https://api.semanticscholar.org/CorpusID:1124694

[132] Tongyu Zhang, Yiran Shen, Guangrong Zhao, Lin Wang, Xiaoming Chen, Lu Bai, and Yuanfeng Zhou. 2024. Swift-Eye: Towards Anti-blink Pupil Tracking for Precise and Robust High-Frequency Near-Eye Movement Analysis with Event Cameras. *IEEE Transactions on Visualization and Computer Graphics* 30, 5 (2024), 2077–2086. doi:10.1109/TVCG.2024.3372039

[133] Xucong Zhang, Yusuke Sugano, and Andreas Bulling. 2019. Evaluation of Appearance-Based Methods and Implications for Gaze-Based Applications. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM. doi:10.1145/3290605.3300646

[134] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. 2015. Appearance-based gaze estimation in the wild. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 4511–4520. doi:10.1109/CVPR.2015.7299081

[135] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. 2017. MPIIGaze: Real-World Dataset and Deep Appearance-Based Gaze Estimation. arXiv:1711.09017 [cs.CV] https://arxiv.org/abs/1711.09017

[136] Ziliang Zhang, Zexin Li, Hyoseung Kim, and Cong Liu. 2024. BOXR: Body and head motion Optimization framework for eXtended Reality. *arXiv preprint arXiv:2410.13084* (2024).

[137] Feng Zheng, Turner Whitted, Anselmo Lastra, Peter Lincoln, Andrei State, Andrew Maimone, and Henry Fuchs. 2014. Minimizing latency for augmented reality displays: Frames considered harmful. In *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 195–200.

[138] Xiaopeng Zhong, Bo Zhang, Amine Bermak, Chi-Ying Tsui, and Man-Kay Law. 2018. A Low-Power Compression-Based CMOS Image Sensor With Microshift-Guided SAR ADC. *IEEE Transactions on Circuits and Systems II: Express Briefs* 65, 10 (2018), 1350–1354. doi:10.1109/TCSII.2018.2859047

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

[140] Pengcheng Zhou, Yan Li, Shuxin Liu, and Yikai Su. 2018. Compact design for optical-see-through holographic displays employing holographic optical elements. *Optics express* 26, 18 (2018), 22866–22876.