EchoVision: Hybrid NPU-CPU Deployment of EfficientViT-SAM and YOLO for Real-Time Assistive Navigation

Su Ho Lim
suho.lim@ucdenver.edu
University of Colorado Denver
USA

Artemis Shaw
arte2837@colorado.edu
University of Colorado Boulder
USA

Huy Dinh
anh-huy.2.dinh@ucdenver.edu
University of Colorado Denver
USA

Nam Bui
nam.bui@ucdenver.edu
University of Colorado Denver
USA

Abstract

Real-time spatial awareness is essential for safe assistive navigation, yet deploying transformer-based vision foundation models on power-constrained mobile devices remains extremely challenging due to fundamental incompatibilities with INT8 fixed-point quantization on neural processing units (NPUs). We present EchoVision, an edge-only auditory navigation aid for visually impaired users that integrates NPU-accelerated YOLOv8n object detection with a hybrid convolutional neural network (CNN)-Central Processing Unit (CPU) deployment of EfficientViT-SAM for selective pixel-level segmentation on a Raspberry Pi 5 with a 26-TOPS Hailo-8 accelerator. Our contributions include (1) a hybrid partitioning strategy executing Stages 0–3 of EfficientViT-SAM on the NPU and attention, neck, and mask decoding on the host CPU, (2) systematic documentation and mitigation of five transformer quantization failure categories (including GELU overflow and MatMul shift-delta errors), and (3) a calibration-based dequantization technique preserving > 0.97 cosine similarity in feature embeddings. The system achieves 16.5–19.7 Frames per Second (FPS) at 13.4 ms detection latency—a 6.6–7.9× throughput gain and 30× latency reduction over the CPU baseline—with selective segmentation averaging 468.6 ms per object.

1 Introduction

Navigating independently remains a critical challenge for individuals with visual impairments [2, 15]. The traditional white cane is limited to the user's physical reach [11], and early Electronic Travel Aids (ETAs) using ultrasonic or infrared sensors [2, 3] detect obstacles but cannot classify them–unable to distinguish a benign wall from an approaching vehicle or a descending staircase.

As shown in Fig. 1, modern vision models close this semantic gap. The You Only Look Once (YOLO) family [16] enables real-time hazard classification, while transformer-based segmentation models like EfficientViT-SAM [20] provide pixel-level spatial understanding. However, deploying either class of model on a portable, battery-operated ETA is non-trivial. On a Raspberry Pi 5 CPU, YOLOv8n alone achieves only 2.5 FPS at 404 ms latency—fundamentally unsafe for navigation. Worse, transformer attention mechanisms (MatMul operations, GELU activations, dynamic tensor reshaping) are incompatible with the INT8 fixed-point arithmetic [6, 12] required by NPUs, and existing compilers fail to quantize them without significant architectural intervention.

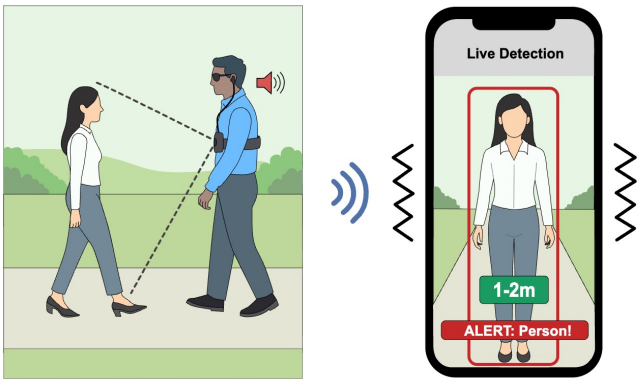


Figure 1: EchoVision overview: Real-time hazard detection and proximity estimation using a hybrid NPU-CPU processing architecture.

We present EchoVision, an edge-only auditory navigation aid for visually impaired users that resolves this tension via a hybrid CNN-transformer deployment on a Raspberry Pi 5 paired with a 26-TOPS Hailo-8 NPU. EfficientViT-SAM is partitioned at the CNN-attention boundary: Stages 0–3 are quantized to INT8 and compiled to the NPU, while Stage 4 attention, the feature pyramid neck, and mask decoder

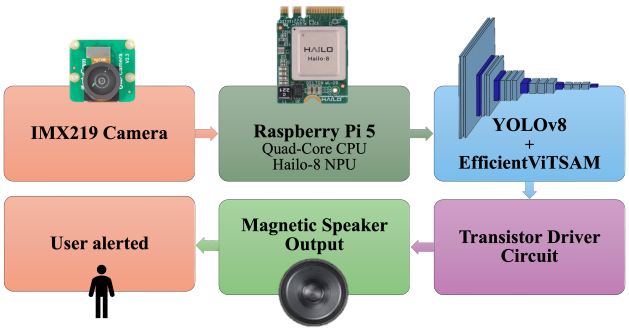


Figure 2: EchoVision system architecture showing hybrid NPU-CPU partitioning of YOLOv8 and EfficientViT-SAM with custom auditory feedback path.

run on the host CPU in FP32. YOLOv8n runs entirely on the NPU, triggering selective SAM segmentation only for high-confidence detections.

In this work, we make the following contributions:

- *Hybrid CNN-Transformer Edge Deployment:* A systematic partitioning methodology for deploying transformer-based vision models on INT8 NPU hardware, demonstrated with EfficientViT-SAM on the Hailo-8.
- *Transformer Quantization Failure Analysis:* Documentation and mitigation of five quantization failure categories–including GELU overflow, attention reshape parsing failures, and MatMul shift-delta incompatibilities–as a practical reference for edge AI practitioners.
- *Two-Model Real-Time Pipeline with Calibration-Based Dequantization:* A YOLO + SAM pipeline achieving 16.5–19.7 FPS at 13.4 ms detection latency—a 6.6–7.9× throughput gain and 30× latency reduction over the CPU baseline—underpinned by a linear regression dequantization mapping that preserves > 0.97 cosine similarity in feature embeddings across the NPU-CPU boundary.

2 Related Work

ETAs relied on ultrasonic and infrared sensors to detect obstacles [2, 3], but lacked semantic understanding for hazard classification. Vision-based ETAs have since emerged as a more powerful alternative [15], with recent systems leveraging deep learning for object detection and scene understanding [17]. However, most still depend on cloud offloading or high-power GPUs, limiting their practicality as portable, battery-operated devices. EchoVision addresses this gap by delivering real-time semantic detection and segmentation entirely on the edge.

The YOLO family [7, 16] has become the standard for real-time object detection on resource-constrained hardware due to its strong accuracy-latency trade-off. The Segment Anything Model (SAM) [8] enabled zero-shot instance segmentation via point or box prompts, but its heavy ViT encoder [4] makes real-time edge inference impractical. EfficientViT-SAM [20] mitigates this by replacing the ViT backbone with a hybrid CNN-transformer architecture [1], significantly reducing compute while maintaining segmentation quality. Our work demonstrates a practical partial NPU deployment of EfficientViT-SAM on commodity edge hardware.

Neural processing units such as the Hailo-8 deliver excellent INT8 performance for CNNs but provide limited support for transformer layers. The Hailo Dataflow Compiler struggles with attention-containing models, including DINOv2 [13], CLIP [14], SegFormer [19], and SwinV2 [10], frequently failing at compilation with errors such as "No valid partition found." We observed similar issues compiling YOLO26n with C2PSA attention blocks. While transformer quantization has been extensively studied for server-grade hardware [6, 12], effective deployment on commodity NPUs remains challenging. EchoVision contributes a pragmatic hybrid partitioning strategy–executing CNN stages on the NPU while running attention, neck, and mask decoding on the CPU—that operates reliably within current toolchain constraints and provides a reusable template for other CNN-transformer architectures on edge hardware.

3 System Architecture

EchoVision operates entirely on the edge as a "digital white cane," integrating two complementary vision models within a unified hardware platform (Fig. 2). The system combines NPU-accelerated object detection with transformer-based segmentation through a hybrid deployment strategy, delivering real-time hazard awareness without cloud dependency.

The core platform consists of a Raspberry Pi 5 (Arm Cortex-A76 quad-core) paired with an IMX219 Camera Module for continuous visual input and a Hailo-8 neural accelerator (26-TOPS) connected via PCIe Gen 3. A CPU-only baseline with YOLOv8n achieved only 2.5 FPS at 404 ms latency, rendering real-time navigation unsafe.

The hybrid pipeline loads two compiled models onto the NPU: YOLOv8n [7] for object detection and the EfficientViT-SAM [20] CNN backbone (Stages 0–3) for feature extraction. The Hailo-8's multi-context allocation enables both models to share the NPU fabric with dynamic switching between network groups. Operations incompatible with INT8 fixed-point arithmetic [6]–including Stage 4 LiteMLA attention [1], the SamNeck feature pyramid, LayerNorm, prompt encoder,

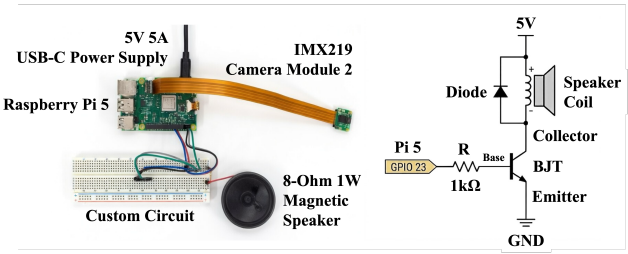


Figure 3: EchoVision hardware prototype (left) and BJT transistor driver circuit schematic (right) for safe high-current speaker actuation.

and mask decoder–run on the host CPU in FP32. This partitioning exploits the natural boundary between convolution-dominated and attention-dominated layers, yielding 16.5–19.7 FPS (a 6.6–7.9× throughput gain and 30× latency reduction over the CPU baseline).

To deliver proximity alerts, standard GPIO pins on the Raspberry Pi 5 are limited to approximately 16 mA, insufficient to drive the 8Ω, 1 W magnetic speaker used for proximity alerts. We mitigate this risk with a custom BJT-based isolation circuit (Fig. 3): a 1kΩ base resistor on GPIO pin 73 controls the transistor, sourcing speaker power directly from the 5V rail with a flyback diode for protection against inductive spikes. We use the `lgpio` backend with `gpiozero`'s `LGPIOFactory` to ensure stable kernel-native PWM on the RP1 I/O controller.

4 System Implementation

4.1 Environment and Model Preparation

We configured 64-bit Raspberry Pi OS with the HailoRT PCIe driver and the `lgpio` backend for kernel-native Pulse Width Modulation (PWM) on the RP1 I/O controller. YOLOv8n [7] was processed through the Hailo Dataflow Compiler (DFC), converting FP32 weights to INT8 [6] and producing a `.hef` file with built-in Non-Maximum Suppression (NMS) post-processing that outputs per-class detection arrays of $[y_1, x_1, y_2, x_2, \text{confidence}]$ directly on the NPU. EfficientViT-SAM [20] required a substantially more complex conversion process, detailed below.

4.2 Transformer Quantization Failures

Deploying EfficientViT-SAM on the Hailo-8 revealed five categories of fundamental incompatibilities between transformer architectures and INT8 fixed-point hardware:

(1) *GELU Activation Overflow:* GELU's polynomial approximation $x \cdot 0.5 \cdot (1 + \tanh(\sqrt{2/\pi} \cdot (x + 0.044715x^3)))$ produces extreme intermediate dynamic range, causing `NegativeSlopeExponentNonFixable` errors where the required shift (35.0) far exceeded the 8 available data bits. Setting 150 layers to `a16_w16` mode reduced the shift to 15.0–exactly matching the bit limit—but the error persisted. We resolved this by replacing all 16 GELU activations with HardSwish ($x \cdot \min(\max(x+3, 0), 6)/6$), which is bounded and quantizes cleanly to INT8 without retraining.

(2) *Attention Reshape Parsing Failures:* EfficientViT-SAM's multi-head attention reshapes tensors from $(B, 256, 256, 32)$ to $(B, 2, 256, 32)$ to distribute across attention heads. The DFC parser incorrectly tracked these reshape operations during the LayerNorm Decomposition phase, producing `AccelerasValueError` shape mismatches between parsed and runtime tensor dimensions.

(3) *MatMul Range Explosion:* The LiteMLA attention mechanism in Stage 4 produces MatMul input ranges of $[0, 5515]$ and $[-1{,}747{,}761,\ 2{,}047{,}653]$—fundamentally incompatible with fixed-point representation. The DFC's `force_range_in` workaround caused downstream activation explosions, with slope values reaching 59.0 against only 8 available data bits.

(4) *ONNX Opset Incompatibility:* PyTorch 2.11's default Open Neural Network Exchange (ONNX) exporter produces opset 18, which the Hailo parser cannot handle—Conv operations omit the `kernel_shape` attribute, causing `IndexError` during translation. We resolved this by forcing the legacy TorchScript exporter via `torch.jit.trace()` and `torch.onnx.utils.export()` at opset 11.

(5) *YOLO26n Compilation Failure:* Despite passing parsing and quantization, YOLO26n failed at compilation with "No valid partition found" errors in layers connected to its C2PSA attention blocks, confirming that the current Hailo-8 toolchain cannot map transformer attention patterns to its dataflow architecture.

4.3 Hybrid Partitioning Strategy

These failures motivated partitioning EfficientViT-SAM-L0 at the CNN-attention boundary. The backbone comprises five stages:

Stage	Block Type	Output Shape
0	ResBlock (Conv, BN)	$1 \times 32 \times 256 \times 256$
1	FusedMBConv (Conv, BN)	$1 \times 64 \times 128 \times 128$
2	FusedMBConv (Conv, BN)	$1 \times 128 \times 64 \times 64$
3	MBConv (Conv, BN)	$1 \times 256 \times 32 \times 32$
4	EfficientViTBlock (LiteMLA)	$1 \times 512 \times 16 \times 16$

Table 1: EfficientViT-SAM-L0 backbone stage decomposition.

Stages 0–3 are pure CNN and, after GELU replacement, compile cleanly to INT8 on the NPU. The modified backbone is exported at opset 11, simplified with `onnxsim` (reducing

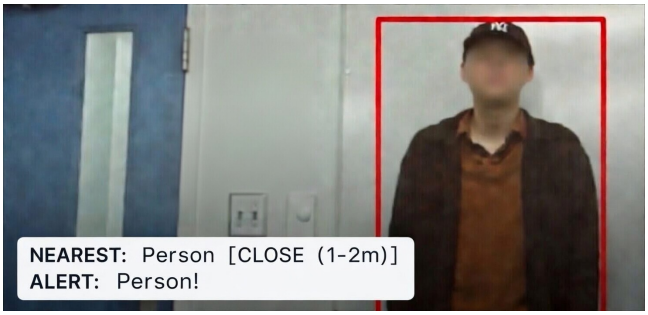


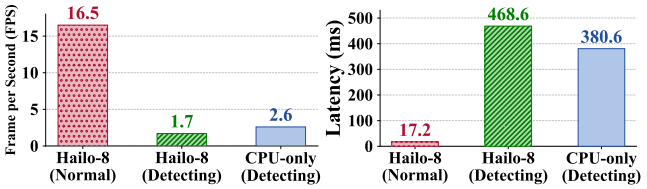
Figure 4: EchoVision output: YOLO detection with SAM segmentation mask and proximity classification.

from 242 to 114 constant nodes), and compiled to a multi-output Hailo Executable Format (HEF) delivering stage2 features (1, 64, 64, 128) and stage3 features (1, 32, 32, 256). Stage 4 (LiteMLA attention), the SamNeck, LayerNorm, prompt encoder, and mask decoder execute on the CPU in FP32.

A critical bridging step is the dequantization mapping from the NPU's uint8 outputs to the CPU model's expected float32 inputs. Simple division by 255 produced incorrect segmentation masks. Using linear regression calibrated on real camera images [9], we derived per-stage scale and offset values (stage2: scale=0.002480, offset=0.337289; stage3: scale=0.832672, offset=101.443069), achieving > 0.97 cosine similarity with pure-PyTorch FP32 reference embeddings.

4.4 Runtime Execution and Data Flow

The pipeline executes in five phases (a representative system output is shown in Fig. 4): (1) Two HEF binaries (YOLOv8n and EfficientViT-SAM CNN backbone) are loaded into the Hailo-8 via the `VDevice` API alongside GPIO subsystem initialization. (2) `picamera2` delivers continuous 640×480 RGB frames to the inference pipeline. (3) The NPU runs YOLOv8n inference averaging 17.2 ms per frame, producing per-class arrays of $[y_1, x_1, y_2, x_2, \text{confidence}]$ via built-in NMS. (4) For each high-confidence detection, the EfficientViT-SAM CNN backbone runs on the NPU (~20 ms), producing stage2 and stage3 feature maps that are dequantized and forwarded to the CPU for Stage 4 attention, neck processing, and mask decoding (~450 ms). Embeddings are cached per frame to amortize backbone cost across multiple detections. (5) Bounding box area ratio ($A = w \times h$ / `frame_area`) is evaluated against empirically tuned thresholds, classifying objects as VERY CLOSE (> 0.30, < 1m), CLOSE (> 0.15, 1–2m), NEAR (> 0.05, 2–3m), MEDIUM (> 0.02, 3–5m), or FAR (> 5m). A decoupled actuation thread drives PWM audio alerts with frequency modulated by proximity class and object type.



(a) Throughput (FPS) **(b) Latency (ms)**

Figure 5: NPU-accelerated Hailo-8 pipeline vs. CPU baseline.

5 Evaluation

5.1 Computational Performance

Three configurations were benchmarked under controlled conditions with consistent scene content and lighting.

CPU-Only Baseline (Test 1): YOLOv8n on the Raspberry Pi 5 CPU yielded 2.5 FPS (range: 1–3) at 404.0 ms mean latency. A separate test pairing YOLO26n with EfficientViT-SAM achieved 2.6 FPS at 380.6 ms—confirming that the CPU bottleneck is architecture-agnostic.

Hailo-8 Detection Only (Test 2): With YOLOv8n fully offloaded to the NPU, the system sustained 19.7 FPS (range: 14.9–21.5) at 13.4 ms mean latency—a 6.6–7.9× throughput gain and 30× latency reduction over the CPU baseline.

Hybrid YOLO + SAM Pipeline (Test 3): The full pipeline operates in two modes. During normal scanning (no SAM triggered), the system achieves 16.5 FPS at 17.2 ms per frame. When SAM segmentation is triggered, throughput drops to 1.7 FPS for a single detected object, with SAM averaging 468.6 ms and peaking at 1,763 ms for complex scenes. As shown in Table 2, worst-case total pipeline latency reaches 1,570.7 ms for one object and 2,546.7 ms for three, with measured FPS of 0.6 and 0.4 respectively—reflecting the CPU-bound transformer components (Stage 4 attention, neck, decoder) that dominate per-frame cost. This bimodal behavior

Pipeline Stage	1 Obj (ms)	2 Obj (ms)	3 Obj (ms)	Hardware
Capture & Preproc	10.7	10.1	11.5	CPU
YOLO Inference	16.5	15.9	17.5	NPU
SAM Stage 0–3	12.8	12.6	12.7	NPU
SAM Stage 4	509.3	467.4	486.2	CPU
SAM Neck/Dec	1021.4	1121.7	2018.8	CPU
Total Latency	**1570.7**	**1627.7**	**2546.7**	–
Measured FPS	**0.6**	**0.6**	**0.4**	–

Table 2: Per-stage latency breakdown for 1–3 objects. FPS reflects the full pipeline including SAM; single-object FPS in Fig. 5 uses average SAM latency of 468.6 ms rather than worst-case total latency.

is by design: continuous high-FPS scanning provides immediate hazard awareness, while SAM is selectively invoked only for high-confidence detections.

5.2 Spatial Awareness and Detection Accuracy

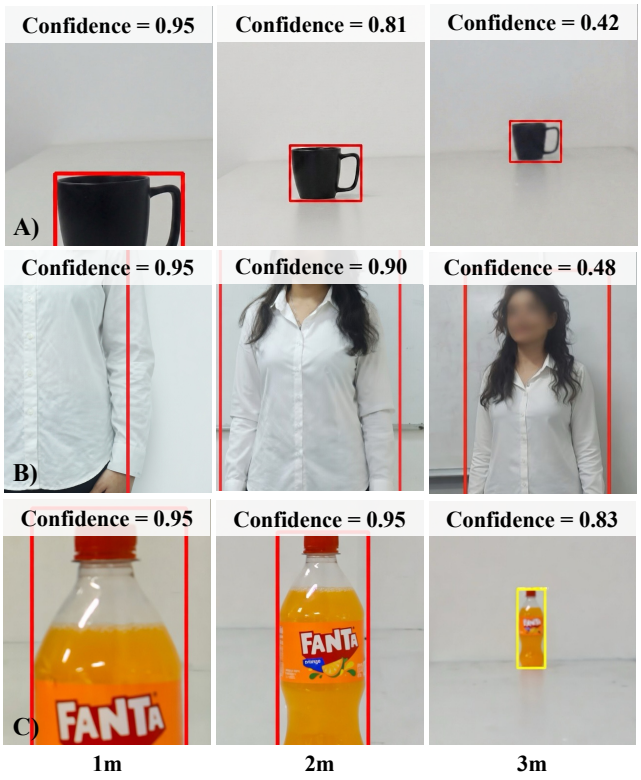


Figure 6: SAM+YOLO detection confidence across distances for Cup (A), Person (B), and Fanta Bottle (C) at 1m, 2m, and 3m.

Fig. 6 and Fig. 7 report YOLOv8n detection confidence across three object classes at varying distances. The Fanta Bottle and Person classes maintain strong confidence at all distances (≥ 0.83 and ≥ 0.48 respectively), while the Cup class degrades sharply at 3m (confidence=0.42), likely due to its small size relative to frame area at distance.

Fig. 8 reports proximity classification accuracy across distance zones. Performance reaches 100% at 2m and beyond, but drops to 30% at 1m. This is an expected limitation of the bounding box area heuristic: at very close range, objects exceed the VERY CLOSE threshold (> 0.30 area ratio) inconsistently due to partial occlusion and frame-edge clipping. Fig. 8 confirms that detection confidence and proximity classification accuracy are correlated at 2m and 3m but diverge

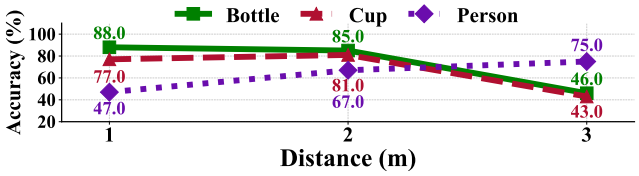


Figure 7: SAM+YOLO detection confidence vs. distance for three object classes (Cup, Person, Bottle) at 1m, 2m, and 3m over n = 80 trials per object.

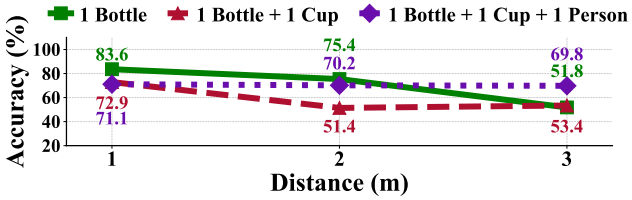


Figure 8: Multi-object detection accuracy of SAM+YOLO across 1 m, 2 m, and 3 m (average accuracy reported when multiple objects are present).

at 1m, where the proximity heuristic fails while YOLO detection remains strong. SAM segmentation masks partially mitigate this by providing actual object silhouettes rather than rectangular bounding boxes for area estimation.

5.3 Quantization Quality

The hybrid partitioning strategy preserves segmentation fidelity despite INT8 quantization of the CNN backbone. Linear regression dequantization calibrated on real camera images [9] yielded the following per-stage results:

- Stage 2: correlation = 0.9816, RMSE = 5.64
- Stage 3: correlation = 0.9946, RMSE = 1,466.4
- End-to-end embedding cosine similarity: 0.9726

The elevated Stage 3 RMSE reflects larger absolute activation magnitudes in deeper layers, but the high correlation confirms that relative feature structure is well-preserved. The > 0.97 cosine similarity between the NPU-CPU hybrid and pure FP32 PyTorch embeddings demonstrates that the partitioning introduces negligible semantic degradation.

6 Discussion

EchoVision shows that transformer-based vision models can be practically deployed on commodity NPU hardware through careful architectural partitioning, while also highlighting the current limitations of this approach.

The fundamental bottleneck is not raw compute but quantization compatibility. The Hailo-8 provides ample 26-TOPS INT8 capacity for the EfficientViT-SAM CNN backbone (~20 ms),

but attention mechanisms generate activation ranges and tensor reshape patterns that existing NPU compilers struggle to map to fixed-point arithmetic. Consequently, the CNN-attention boundary emerges as a natural and effective partitioning point: convolutional layers produce constrained activations well-suited to INT8 [5, 18], whereas attention layers perform unbounded query-key dot products that resist effective quantization.

The bimodal performance profile is well-suited to assistive navigation. Continuous YOLO scanning at 16.5 FPS delivers immediate hazard awareness, while selective SAM segmentation at ∼1.7 FPS is invoked only for high-confidence detections when detailed spatial information is needed. Compared to the CPU-only baseline (2.6 FPS), the NPU provides a 6.6–7.9× improvement in scanning throughput. Notably, SAM latency remains largely CPU-bound in both configurations, indicating that the main benefit of NPU offloading is freeing CPU resources for the attention and decoder stages rather than dramatically reducing end-to-end segmentation time.

Limitations. Several constraints should be acknowledged. First, substituting GELU with HardSwish introduces a distributional shift in the CNN backbone features. Although our measurements show high cosine similarity (> 0.97), a thorough evaluation against ground-truth segmentation masks is still needed. Second, the uint8-to-float32 dequantization currently relies on scene-specific linear regression coefficients; extracting parameters directly from HEF metadata would be more robust. Finally, the monocular proximity heuristic struggles to distinguish small nearby objects from larger distant ones using bounding box area alone; using stereo vision would improve robustness.

Future Work. Quantization-aware training of the attention layers could enable full NPU deployment and eliminate the CPU bottleneck. Future NPU architectures (e.g., Hailo-10H and Hailo-15) are expected to offer broader transformer support. A compact wearable enclosure with integrated cooling would address the observed thermal limitations. More generally, the hybrid partitioning approach presented here is applicable to any CNN-transformer hybrid model and provides a practical deployment template while NPU toolchains evolve toward full transformer support.

7 Conclusion

We presented EchoVision, a fully edge-based assistive navigation system that demonstrates how transformer-based vision models can be practically deployed on commodity NPU hardware through architectural partitioning. By splitting EfficientViT-SAM at the CNN-attention boundary, the system achieves real-time YOLOv8n detection at 16.5–19.7 FPS while delivering selective high-quality segmentation at an average of 468.6 ms per object.

This work contributes a systematic analysis of five transformer quantization failure modes on the Hailo-8 and shows that hybrid NPU-CPU partitioning is an effective, immediately deployable strategy when full transformer support is unavailable. The approach is applicable to any CNN-transformer architecture and provides a practical template as NPU toolchains continue to evolve. EchoVision takes a concrete step toward bringing advanced semantic vision to wearable assistive devices for the visually impaired.

References

[1] Han Cai, Junyan Li, Muyan Hu, Chuang Gan, and Song Han. 2023. Efficientvit: Lightweight multi-scale attention for high-resolution dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*. 17302–17313.

[2] Piyush Chanana, Rohan Paul, M Balakrishnan, and PVM Rao. 2017. Assistive technology solutions for aiding travel of pedestrians with visual impairment. *Journal of rehabilitation and assistive technologies engineering* 4 (2017), 2055668317725993.

[3] Dimitrios Dakopoulos and Nikolaos G Bourbakis. 2009. Wearable obstacle avoidance electronic travel aids for blind: a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 40, 1 (2009), 25–35.

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[6] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. 2018. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2704–2713.

[7] Glenn Jocher, Jing Qiu, and Ayush Chaurasia. 2023. *Ultralytics YOLO*. https://github.com/ultralytics/ultralytics

[8] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*. 4015–4026.

[9] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*. Springer, 740–755.

[10] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. 2022. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 12009–12019.

[11] Mohamed Dhiaeddine Messaoudi, Bob-Antoine J Menelas, and Hamid Mcheick. 2022. Review of navigation assistive tools and technologies for the visually impaired. *Sensors* 22, 20 (2022), 7888.

[12] Markus Nagel, Marios Fournarakis, Rana Ali Amjad, Yelysei Bondarenko, Mart Van Baalen, and Tijmen Blankevoort. 2021. A white paper on neural network quantization. *arXiv preprint arXiv:2106.08295* (2021).

[13] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. 2023. Dinov2: Learning robust visual

features without supervision. *arXiv preprint arXiv:2304.07193* (2023).

[14] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PmLR, 8748–8763.

[15] Santiago Real and Alvaro Araujo. 2019. Navigation systems for the blind and visually impaired: Past work, challenges, and open problems. *Sensors* 19, 15 (2019), 3404.

[16] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 779–788.

[17] Dillon Reis, Jordan Kupec, Jacqueline Hong, and Ahmad Daoudi. 2023. Real-time flying object detection with YOLOv8. *arXiv preprint arXiv:2305.09972* (2023).

[18] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4510–4520.

[19] 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.

[20] Zhuoyang Zhang, Han Cai, and Song Han. 2024. EfficientViT-SAM: Accelerated segment anything model without accuracy loss. *arXiv preprint arXiv:2402.05008* (2024).