PRIMM: Perception Using Integrated Multi-Modal Modularity

Lily Lamb
University of South Carolina
lrlamb@email.sc.edu

Mohammadreza Mohammadi
University of South Carolina
mohammm@email.sc.edu

Ramtin Zand
University of South Carolina
ramtin@cse.sc.edu

Abstract

Embedded vision systems face a fundamental trade-off between high-resolution sensing and limited edge resources. In this paper, we propose the PRIMM system, which addresses this challenge through adaptive, multi-modal perception that preserves efficiency while improving robustness. Unlike camera-only pipelines that degrade in low-light or occluded scenes, PRIMM fuses RGB imagery with LiDAR-derived dense depth maps generated near the sensor. Depth maps are constructed through LiDAR-to-camera transformation followed by parallelized chunk-based interpolation, enabling efficient preprocessing on near-sensor compute units. Selective sensor activation further reduces redundant workload by engaging LiDAR only when visual confidence is low. Evaluation across multiple datasets shows that PRIMM improves perception robustness and detection accuracy with minimal impact on energy consumption and latency, demonstrating the effectiveness of near-sensor multimodal fusion for adaptive embedded vision.

1. Introduction

Advances in CMOS image sensors (CISs) have enabled compact, high-resolution, and energy-efficient cameras, forming the backbone of modern embedded vision systems [8]. Despite these improvements, embedded vision faces two persistent challenges: the high data rates of modern sensors generate large volumes of information, and resource-constrained edge processors must execute increasingly complex machine learning (ML) workloads in real time [1]. Continuous data transfers between the sensor and processing units dominate both energy consumption and latency, creating a bottleneck that limits the performance of real-time applications such as autonomous navigation and high-security monitoring [5].

Prior work has explored in-sensor and near-sensor computing to reduce energy-intensive data transfers and offload early-stage processing [2–4, 7, 9–11]. These approaches optimize end-to-end embedded vision pipelines by pushing computation closer to the sensor, improving both energy efficiency and latency for object recognition and tracking tasks. However, camera-only perception remains vulnerable under visually challenging conditions such as low light, occlusion, and perceptual ambiguity.

To overcome these limitations, multi-sensor fusion has emerged as a robust paradigm that integrates complementary sensing modalities, most commonly RGB cameras and LiDAR [6]. LiDAR contributes dense, metric-scale depth measurements that enable precise object localization and reliable scene understanding even in adverse illumination or weather. Here, we propose the PRIMM system, which builds on near-sensor computing principles to enable depth-aware multimodal perception, performing LiDAR-to-camera transformation and camera-to-image projection within near-sensor hardware. The resulting dense depth maps are fused with RGB imagery to support robust and efficient multimodal object detection.

2. PRIMM Architecture

Camera-only systems inherently struggle with object detection in visually ambiguous or low-light environments due to their reliance on appearance-based cues. To address these limitations, multi-sensor fusion has become a standard approach, integrating complementary data from varying modalities such as RGB cameras and LiDAR [6]. While such fusion improves robustness and situational awareness, it also introduces notable challenges, including higher hardware costs, increased power consumption, and greater computational complexity.

PRIMM mitigates these challenges through selective sensor activation, dynamically engaging additional modalities only when necessary. In particular, LiDAR is incorporated to provide precise three-dimensional spatial information that complements RGB imagery. Its dense metric-scale depth measurements enable accurate object localization and scene understanding, even in adverse illumination or weather conditions where passive sensors perform poorly.

The PRIMM's modular design facilitates efficient runtime adaptation: LiDAR can be deactivated during high-

Algorithm 1 Dense Map with Parallelized Chunk Processing

function GENERATE_DEPTHMAP(Pts, n, m, $grid$)
  $ng \leftarrow 2 \cdot grid + 1$
  Initialize mX, mY, mD matrices of size (m, n) with appropriate initial values
  $mX[Pts[1], Pts[0]] \leftarrow Pts[0] - \text{round}(Pts[0])$
  $mY[Pts[1], Pts[0]] \leftarrow Pts[1] - \text{round}(Pts[1])$
  $mD[Pts[1], Pts[0]] \leftarrow Pts[2]$
  Initialize KmX, KmY, KmD matrices of size $(ng, ng, m - ng, n - ng)$
  Initialize S, Y matrices of size $(m - ng, n - ng)$ with zeros
  Parallelize: Execute PROCESS_CHUNK(i, j) for each (i, j) in the grid using ThreadPoolExecutor
  for all processed chunks **do**
    Update Y and S with the results from the chunk
  end for
  Set all zeros in S to 1 to avoid division by zero
  Initialize out matrix of size (m, n)
  $out[grid+1 : m-grid,\ grid+1 : n-grid] \leftarrow \frac{Y}{S}$
  return out
end function

function PROCESS_CHUNK(i, j)
  Compute $mx \leftarrow$ slice of mX based on (i, j)
  Compute $my \leftarrow$ slice of mY based on (i, j)
  Compute $s \leftarrow \frac{1}{\sqrt{mx^2+my^2}}$
  Compute $Y_{\text{chunk}} \leftarrow s \cdot$ slice of mD based on (i, j)
  return Y_{chunk}, s
end function

confidence visual tracking, then re-engaged dynamically when environmental cues indicate degraded camera performance. Through this selective integration, PRIMM achieves a balanced trade-off between perceptual robustness and computational efficiency, maintaining high detection accuracy in visually challenging scenes while minimizing redundant processing across modalities. Figure 1 illustrates the block diagram of the PRIMM system, where sparse LiDAR point clouds are first converted into dense depth maps within a near-sensor microprocessor unit (MPU) (step ❶), then combined with RGB data and transmitted to the edge AI hardware (step ❷) for multimodal perception and inference (step ❸).

2.1. PRIMM Implementation Details

We implement the PRIMM system using a heterogeneous architecture. We utilize a microprocessor with two ARM A57 cores operating at 0.9 GHz as the near-sensor processing unit. Communication between the sensing system and the microprocessor is handled via the PCIe 4.0 protocol

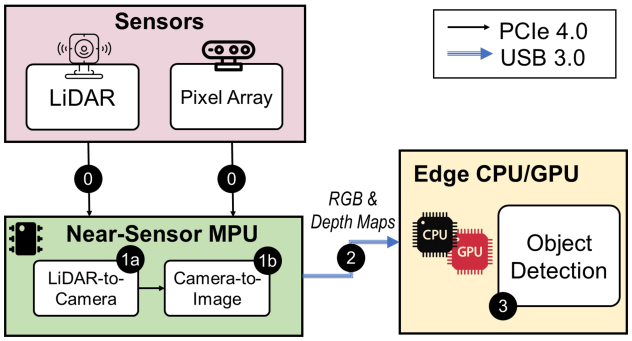

Figure 1. Block diagram of the PRIMM system architecture for multimodal object detection. In this setup, the near-sensor MPU performs the LiDAR-to-camera transformation and camera-to-image projection, while the edge AI hardware executes the multimodal object detection task.

[12], which provides a bandwidth of 16 Gbps and consumes approximately 10^{-11} joules per bit. Additionally, we use the Jetson Nano as the primary embedded system processor. The Jetson Nano features a quad-core ARM A57 processor, four 32-CUDA cores, and 4 GB of memory. Communication between the Jetson Nano and near-sensor microprocessor is facilitated through USB 3.0, which offers a bandwidth of 5 Gbps and consumes 10^{-7} joules per bit.

2.2. Pre-Processing

As shown in Fig. 1, LiDAR integration within the PRIMM system begins with a dedicated pre-processing stage implemented in the near-sensor MPU, which ensures spatial and temporal alignment between the sensing modalities. The data is structured for efficient multi-modal processing by enforcing precise spatial alignment between RGB images and Velodyne LiDAR point clouds. This preparation pipeline consists of two PRIMMary steps: (1) LiDAR-to-camera transformation (step ❶a), which aligns LiDAR data with the camera's coordinate system using precomputed rotation and translation matrices, and (2) camera-to-image projection (step ❶b), which maps 3D LiDAR points onto the 2D image plane through the application of intrinsic and extrinsic calibration parameters.

Following LiDAR-camera alignment, Algorithm (1) generates dense depth maps from sparse LiDAR point clouds via grid-based interpolation with distance-weighted averaging. The algorithm takes as input the LiDAR points (`Pts`), the target depth map dimensions (`n`, `m`), and a `grid` parameter controlling sparsity. It initializes offset and depth matrices (`mX`, `mY`, `mD`) and populates them with input point data. Precomputed slices and intermediate accumulation matrices (`S`, `Y`) enable efficient computation. Interpolation is performed by the `process_chunk` function, which applies inverse distance weighting over each grid chunk.

To accelerate computation, the algorithm parallelizes chunk processing using Python's `ThreadPoolExecutor`, distributing workload across multiple MPU threads. After aggregation, zero entries in `S` are replaced with ones to prevent division by zero, and the final depth map is computed by dividing `Y` by `S`.

This implementation addresses the inefficiency of traditional nested-loop approaches by combining loop refactoring, multithreading, and memory slicing, achieving a significant reduction in runtime without sacrificing precision. The resulting dense depth maps are well-suited for multimodal fusion, allowing GPU resources to remain available for concurrent tasks such as high-resolution object detection, making this approach particularly effective for the PRIMM framework.

3. Results and Discussions

We explore three fusion strategies, early, mid, and late fusion, that differ in how and when the modalities are integrated within the network. In the *early fusion* approach, RGB and depth data are combined at the input stage, enabling the network to learn joint low-level spatial and geometric features from the outset. The *mid fusion* strategy employs separate modality-specific encoders whose intermediate feature maps are merged within the backbone, allowing the model to capture richer cross-modal correlations while preserving modality-specific representations. The late fusion approach processes each modality independently until the detection head, where feature maps are combined for final object prediction.

Experiments are performed on the KITTI dataset, which offers synchronized RGB images and LiDAR point clouds. We train and evaluate three YOLOv8 variants tailored for multimodal perception, implementing early, mid, and late fusion strategies. Model accuracy is reported using mAP@50 and mAP@50–95 metrics, while inference latency is measured across a GPU-accelerated Jetson Nano. All models are implemented in PyTorch and exported to ONNX for deployment. Due to memory and compute limitations, models were trained on both compressed lower resolution (256×96) and high-resolution images (1280×384) for deployment consideration.

3.1. Accuracy

On the KITTI dataset, the RGB-only baseline achieves 88.3% mAP@50 and 63.6% mAP@50–95 at high resolution. As shown in Table 1, integrating depth information yields consistent accuracy improvements, with mid-fusion attaining the highest performance at 91.7% mAP@50 and 69.3% mAP@50–95. Late fusion performs comparably (91.6% and 69.6%), slightly outperforming mid-fusion in mAP@50–95, suggesting enhanced localization precision under stricter IoU thresholds. These results demonstrate

Table 1. The impact of different configurations and fusion levels on accuracy.

Modality Fusion	Original Images 1280×384		Compressed Images 256×96	
	mAP@50 (%)	mAP@50-95 (%)	mAP@50 (%)	mAP@50-95 (%)
RGB	88.3	63.6	69.9	46.9
RGB-D-Early	90.4	67.4	45.5	23.8
RGB-D-Mid	91.7	69.3	51.6	25.2
RGB-D-Late	91.6	69.6	51.5	26.2

that LiDAR-derived depth maps provide complementary geometric cues that strengthen RGB-based perception, particularly under visually ambiguous or low-contrast conditions. This aligns with PRIMM's design objective of selectively integrating LiDAR features to improve spatial reasoning while maintaining computational efficiency.

Results from the low-resolution inputs generaed by a 4×4 compression reveal a substantial accuracy degradation across all fusion strategies, underscoring the importance of high-resolution perception for reliable detection. Lower spatial detail in the 256×96 inputs limits the model's ability to capture fine-grained geometric and textural cues, particularly for smaller or distant objects. These findings highlight that optimal performance requires inference on high-resolution imagery, a capability that PRIMM supports in real time through its pipelined execution and near-sensor pre-processing framework.

3.2. Latency

The latency analysis of the PRIMM is summarized in Table 2. The table presents the latency breakdown for different multimodal fusion strategies implemented on PRIMM, compared against the baseline. Latency measurements were collected and averaged over 200 iterations for each model. The RGB-only YOLOv8 achieved an average inference latency of 49 ms per frame. Incorporating depth information increased computational complexity, resulting in latencies of 53 ms for early fusion, 90 ms for mid fusion, and 95 ms for late fusion. All RGB–Depth configurations required an additional 23 ms for pre-processing to convert raw LiDAR point clouds into dense depth maps.

In the conventional baseline, both RGB and LiDAR raw data are transferred to the edge AI hardware (CPU–GPU), where pre-processing and multimodal inference are performed. In contrast, the PRIMM pipeline executes near-sensor pre-processing on the embedded MPU prior to inference, running concurrently with inference on the previous frame. This stage converts raw LiDAR point clouds into dense depth maps locally, significantly reducing the volume of data transmitted to the edge AI hardware as the pre-processed depth maps are approximately 19× smaller than the raw point clouds data. As a result, PRIMM substantially lowers communication overhead by avoiding redun-

Table 2. Latency breakdown (in milliseconds) of various multimodal fusion strategies.

System		Fusion Strategy		
		Early	Mid	Late
Baseline	Sens→Proc Transfer		5.1	
	Processing	76	113	118
	End-to-End	81.1	118.1	123.1
PRIMM	Sens→MPU Transfer		1.59	
	MPU Pre-processing		23	
	MPU→Proc Transfer		2.35	
	Processing	53	90	95
	End-to-End	79.94	116.94	121.94

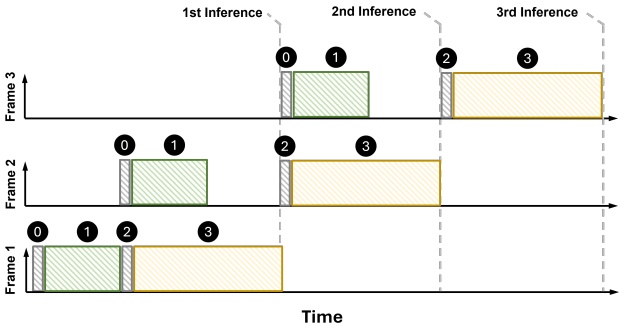

Figure 2. Pipelined execution structure of the PRIMM system for multimodal object detection. The near-sensor MPU executes the LiDAR-to-camera transformation and camera-to-image projection, while the edge AI hardware performs multimodal object detection. The pipeline organization enables concurrent execution of preprocessing and inference tasks across frames, thereby increasing system throughput and effectively hiding preprocessing latency. Note that the timing ratios between data transfer, inference, and preprocessing steps do not reflect the actual system timing and are included only to illustrate the pipeline concept.

Table 3. Throughput analysis (in FPS) of PRIMM compared to the conventional baseline for multimodal object detection.

System	Fusion Strategies		
	Early	Mid	Late
Baseline	12.33	8.47	8.12
PRIMM	18.07	10.83	10.27

dant high-bandwidth data transfers between the sensors and the edge AI hardware.

In addition to reducing data movement between sensing and compute domains, PRIMM's heterogeneous processing architecture enables a pipelined execution framework, as illustrated in Fig. 2. By overlapping the pre-processing of frame n with the inference of frame $n-1$, PRIMM effectively conceals pre-processing latency and enhances real-time responsiveness. This pipelined design results in throughput gains of 46%, 27%, and 26% (in FPS) for early, mid, and late fusion strategies, respectively, compared to conventional systems, as summarized in Table 3.

Overall, these results demonstrate that PRIMM's selective multimodal integration and pipelined execution framework jointly deliver substantial efficiency gains without compromising perception accuracy. By strategically managing sensing, computation, and communication across heterogeneous hardware, PRIMM achieves a balanced design that scales effectively from single-sensor operation to multimodal perception under dynamic real-world conditions.

4. Conclusion

This work presents the PRIMM vision system, which integrates LiDAR-based depth perception with RGB imagery for embedded object detection. By performing LiDAR-to-camera transformation and dense depth map generation near the sensor, the system achieves precise spatial alignment while minimizing data transfer and computational load. Parallelized processing of LiDAR chunks and selective sensor activation further optimize latency and energy consumption, ensuring efficient operation on resource-constrained edge hardware. Experimental results demonstrate that PRIMM improves detection robustness under challenging conditions, including low-light and occluded scenes, without significantly increasing energy or latency. These results highlight the effectiveness of combining near-sensor processing with multimodal fusion, providing a practical pathway for accurate, adaptive, and energy-efficient perception in modern embedded vision systems.

Acknowledgments

This work is supported by the National Science Foundation (NSF) under grant number 2340249.

References

[1] Jaehyuk Choi, Seokjun Park, Jihyun Cho, and Euisik Yoon. An energy/illumination-adaptive cmos image sensor with reconfigurable modes of operations. *IEEE Journal of Solid-State Circuits*, 50(6):1438–1450, 2015. 1

[2] Piotr Dudek, Thomas Richardson, Laurie Bose, Stephen Carey, Jianing Chen, Colin Greatwood, Yanan Liu, and Walterio Mayol-Cuevas. Sensor-level computer vision with pixel processor arrays for agile robots. *Science Robotics*, 7(67): eabl7755, 2022. 1

[3] Robert J Gove. Cmos image sensor technology advances for mobile devices. In *High Performance Silicon Imaging*, pages 185–240. Elsevier, 2020.

[4] Tzu-Hsiang Hsu, Yi-Ren Chen, Ren-Shuo Liu, Chung-Chuan Lo, Kea-Tiong Tang, Meng-Fan Chang, and Chih-Cheng Hsieh. A 0.5-v real-time computational cmos image sensor with programmable kernel for feature extraction. *IEEE Journal of Solid-State Circuits*, 56(5):1588–1596, 2020. 1

[5] Rasheed Hussain and Sherali Zeadally. Autonomous cars: Research results, issues, and future challenges. *IEEE Communications Surveys & Tutorials*, 21(2):1275–1313, 2018. 1

[6] Lily Lamb, Mohammadreza Mohammadi, and Ramtin Zand. Multi-modal vision at the edge: Toward low-latency perception for autonomous systems. In *2025 IEEE International Conference on Omni-layer Intelligent Systems (COINS)*, pages 1–7. IEEE, 2025. 1

[7] Jiajun Li, Yi Luo, Reza Molavi, and Shahriar Mirabbasi. A computational cmos image sensor architecture using in-pixel pwm-sci-based mac with reconfigurable kernel size. In *2024 22nd IEEE Interregional NEWCAS Conference (NEWCAS)*, pages 198–202. IEEE, 2024. 1

[8] Sunetra K Mendis, Sabrina E Kemeny, Russell C Gee, Bedabrata Pain, Craig O Staller, Quiesup Kim, and Eric R Fossum. Cmos active pixel image sensors for highly integrated imaging systems. *IEEE Journal of Solid-State Circuits*, 32 (2):187–197, 1997. 1

[9] Mohammadreza Mohammadi, Mehrdad Morsali, Sepehr Tabrizchi, Brendan Reidy, Arman Roohi, Shaahin Angizi, and Ramtin Zand. Pixelprune: Optimizing aiot vision systems via in-sensor segmentation and adaptive data transfer. In *Proceedings of the Great Lakes Symposium on VLSI 2025*, pages 312–319, 2025. 1

[10] Brendan Reidy, Sepehr Tabrizchi, Mohammadreza Mohammadi, Shaahin Angizi, Arman Roohi, and Ramtin Zand. Hirise: High-resolution image scaling for edge ml via in-sensor compression and selective roi. In *Proceedings of the 61st ACM/IEEE Design Automation Conference*, pages 1–6, 2024.

[11] Xu Ren, Liqiao Liu, Yandong He, and Gang Du. A dual-mode cmos image sensor based on in-pixel frame differencing. In *2024 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE, 2024. 1

[12] Debendra Das Sharma. Pci-express: Evolution of a ubiquitous load-store interconnect over two decades and the path forward for the next two decades. *IEEE Circuits and Systems Magazine*, 24(2):47–61, 2024. 2