ORB Feature Extraction on Embedded Platforms: A Heterogeneous CPU-GPU-PVA Approach

Hamid Moghadaspour*, Nuno Neves†, Oscar Ferraz*, Gabriel Falcao*

*Instituto de Telecomunicações, Department of Electrical and Computer Engineering,
University of Coimbra, 3030-290 Coimbra, Portugal
{hamid.moghadaspour, oscar.ferraz, gff}@co.it.pt

†INESC-ID, Instituto Superior Técnico, University of Lisbon, 1000-019 Lisbon, Portugal
nuno.neves@inesc-id.pt

***Abstract*—In this work, we present a hybrid Oriented FAST and Rotated BRIEF (ORB) feature extraction pipeline that integrates the CPU, GPU, and NVIDIA Jetson's Programmable Vision Accelerator (PVA) to optimize performance on embedded platforms. By assigning each stage of the pipeline to the most suitable processing unit, we reduce latency while preserving feature quality. Specifically, we replace the conventional Features From Accelerated Segment Test (FAST) detector with a Harris corner detector, achieving lower detection time and potentially enhanced localization accuracy. Experiments on Jetson AGX Xavier show that our hybrid design achieves a 1.16× speedup and 13% energy savings over the CPU+GPU setup, lowering energy use from 0.203 to 0.199 μJ/pixel. The pipeline remains SLAM-compatible and highlights the potential of heterogeneous scheduling for efficient, real-time visual processing on embedded systems.**

I. Introduction

ORB-SLAM3, a popular Visual Simultaneous Localization And Mapping (VSLAM) framework, excels in accuracy and real-time performance across diverse systems. The front-end feature extraction module employs the Oriented FAST and Rotated BRIEF (ORB) technique, which is highly efficient and repeatable and, making it excellent for real-time applications. The Features From Accelerated Segment Test (FAST) detector can cause GPU overload and poor corner localization, limiting low-power embedded devices.

Despite its superior feature detection accuracy over FAST [1], [2], the Harris corner detector is rarely employed on embedded edge devices due to its computing needs. This work utilized Programmable Vision Accelerator (PVA), which PVA is a low-power, fixed-function hardware block on the NVIDIA Jetson AGX Xavier platform. It was created to offload CPU and GPU operations like convolutional filtering, corner detection, and object tracking. Our solution uses the NVIDIA Vision Programming Interface (VPI) library, which provides a high-level API for the PVA-implemented Harris detector without manual kernel development. This hybrid solution uses a Harris corner detector instead of GPU-based FAST corner detection, while remaining compatible with the ORB feature extraction pipeline.

II. Related Work

Several studies tackled Simultaneous Localization And Mapping (SLAM)'s performance limitations and recommended CUDA-based acceleration for critical modules [3]. Muzzini et al. [4] provide a full evaluation of GPU-accelerated ORB feature extraction. Similar projects, such as cuVSLAM [5] and RPG Zürich [6], show considerable speedups by GPU optimization.

A historical overview of hybrid SLAM platforms is presented in [7], [8], emphasizing the growing relevance of heterogeneous computing in real-time robotics. Hardware-based acceleration has also been extensively studied. Field-Programmable Gate Arrays (FPGA) implementations, such as those in [9]–[12], boost ORB pipeline throughput and energy efficiency significantly. Hybrid CPU-GPU-FPGA platforms are promising, as seen in [13], [14]. Compared to FAST, the Harris corner detection algorithm is more computationally demanding due to its reliance on intensity gradients and eigenvalue analysis of the second-moment matrix within a sliding window [2], [15]. While FAST uses pixel intensity comparisons around a circle and simple thresholding [6], Harris computes image gradients, squares them, and applies a Gaussian filter to obtain the structure tensor [15]. This results in better robustness under noise or small motions, but at a higher computational cost.

TABLE I
Execution Time Breakdown of ORB Pipeline Across Configurations

Stage	CPU (1 thread)	CPU (8 thread)	CPU+GPU	CPU+GPU+PVA
Pyramid	10.8 ms	7.2 ms	9.3 ms	9.3 ms
Blurring	17.6 ms	13.5 ms	11.1 ms	11.1 ms
FAST	31.3 ms	4.3 ms	9.1 ms	-
Harris	-	-	-	**3.9 ms (PVA)**
Octree (in CPU)	18.9 ms	18.9 ms	18.9 ms	18.9 ms
Orientation	5.4 ms	5.4 ms	3.2 ms	3.2 ms
Descriptor	15.2 ms	15.2 ms	6.3 ms	6.3 ms
Total (ms)	99.2	64.5	39_{GPU} 18.9_{CPU}	**33.8_{GPU}** 18.9_{CPU} 3.9_{VPA}
fps_max	10.08	15.5	25.6	**29.6**
Energy Consumption Index μJ/pixel	0.518	0.376	0.203	**0.199**

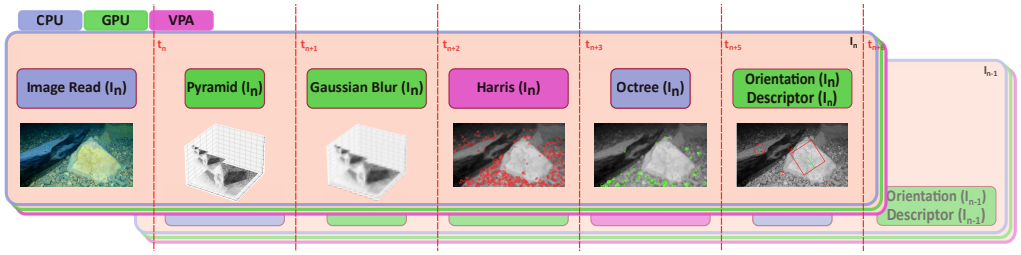

Fig. 1. Pipelined execution of ORB stages across CPU, GPU, and PVA, enabling parallel processing of multiple frames with throughput limited by the CPU-based Octree stage.

III. Hybrid Implementation Strategy and Experimental Results

The ORB [16] pipeline includes image pyramid generation, corner detection (typically FAST), Gaussian blurring, orientation estimation, BRIEF descriptor computation [14], and spatial filtering via Octree. As shown in Fig. 1, we map these steps to a heterogeneous execution strategy: the CPU handles image read and Octree filtering; the GPU performs pyramid construction and Gaussian blurring, orientation estimation, and descriptor extraction; and the PVA extract Harris corners. This hybrid design, unlike prior CPU-only [**?**] or GPU-only [3] approaches, enables pipelined execution of consecutive frames, increasing throughput by overlapping computation across platforms.

Table I compares execution times across four configurations. Although the total frame time does not significantly drop in the hybrid case, system throughput improves due to effective parallelization across computing units. A key enhancement is delegating Harris corner detection to the PVA. This was motivated by the availability of a well-optimized Harris implementation via NVIDIA's VPI and the reduction in latency, from 9.1 ms (GPU-FAST) to 3.9 ms (PVA-Harris). Fig. 1 shows the system adopts a streaming pipeline where different images are processed concurrently at different stages. Consequently, overall throughput is limited by the slowest stage, which is Octree filtering on the CPU at 18.9 ms, rather than the total sequential time. Despite this bound, the hybrid pipeline reaches a peak rate of 29.6 fps with improved parallel efficiency. This configuration also yields the lowest energy consumption index at 0.199 μJ/Pixel, confirming enhanced resource utilization and energy-aware performance compared to other configurations. In summary, our hybrid approach reduces stage-level latency and illustrates how heterogeneous scheduling and pipelining can significantly enhance SLAM performance on embedded systems.

IV. Conclusion and Future Work

We presented a hybrid ORB pipeline that takes advantage of the CPU, GPU, and PVA capabilities available in embedded SoCs. By delegating Harris corner detection to the PVA, we lowered latency at that stage while potentially boosting keypoint accuracy over standard FAST. Although Octree filtering is CPU-bound due to data dependencies, its impact is mitigated by partial GPU overlap. Parallel execution across heterogeneous units enhances throughput without requiring hardware changes.

To solve the remaining obstacles, future work will look into dynamic task scheduling under power limits, as well as additional parallelization of Octree filtering.

Acknowledgment This work was supported by Instituto de Telecomunicações and INESC-ID by Fundação para a Ciência e a Tecnologia (FCT) by project reference 2022.06780.PTDC, 2022.04020.PTDC, LA/P/0083/2020, UIDP/50009/2020, UIDB/50008/2020, and UIDB/50021/2020.

References

[1] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *European conference on computer vision*. Springer, 2006, pp. 430–443.
[2] Zhao *et al.*, "DTFS-eHarris: a high accuracy asynchronous corner detector for event cameras in complex scenes," *Applied Sciences*, 2023.
[3] A. Mamri, M. Abouzahir, M. Ramzi, and R. Latif, "ORB-SLAM accelerated on heterogeneous parallel architectures," *E3S Web of Conferences*, vol. 229, p. 01055, 2021, publisher: EDP Sciences.
[4] Muzzini *et al.*, "Brief announcement: Optimized gpu-accelerated feature extraction for orb-slam systems," in *Proceedings of the 35th ACM Symposium on Parallelism in Algorithms and Architectures*, 2023.
[5] A. E. Alexander Korovko, Dmitry Slepichev, "cuVSLAM: CUDA accelerated visual odometry and mapping," *arXiv*, 2025. [Online]. Available: https://doi.org/10.48550/arXiv.2506.04359
[6] D. Nagy and et al., "Faster than FAST: GPU-Accelerated Frontend for High-Speed VIO," *RPG Zürich*, 2020.
[7] Chen *et al.*, "SLAM overview: From single sensor to heterogeneous fusion," *Remote Sensing*, 2022.
[8] Song *et al.*, "Mis-slam: Real-time large-scale dense deformable slam system in minimal invasive surgery based on heterogeneous computing," *IEEE Robotics and Automation Letters*, 2018.
[9] A. Costa, P. Tomás, N. Roma, and N. Neves, "Real-Time ORB Accelerator with ROS Integration for Embedded FPGA SoCs," in *2025 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2025.
[10] Wasala *et al.*, "An efficient real-time FPGA-based ORB feature extraction for an UHD video stream for embedded visual SLAM," *Electronics*, 2022.
[11] Taranco *et al.*, "LOCATOR: Low-power ORB accelerator for autonomous cars," *Journal of Parallel and Distributed Computing*, 2023.
[12] V. Vemulapati and D. Chen, "ORB-based SLAM accelerator on SoC FPGA," *arXiv*, 2022. [Online]. Available: https://arxiv.org/abs/2207.08405
[13] a. o. Kim, "SuperNoVA: Algorithm-Hardware Co-Design for Resource-Aware SLAM," pp. 1035–1051, 2025.
[14] Mamri *et al.*, "ORB-SLAM accelerated on heterogeneous parallel architectures," in *E3S Web of Conferences*. EDP Sciences, 2021.
[15] Y. Wang, Y. Li, J. Wang, H. Lv, and Z. Yang, "A Target Corner Detection Algorithm Based on the Fusion of FAST and Harris," *Mathematical Problems in Engineering*, 2022.
[16] Rublee *et al.*, "ORB: An efficient alternative to SIFT or SURF," in *2011 International Conference on Computer Vision*, 2011.