Autonomous Monocular Navigation for Robotic Vehicles using ESP32-CAM and TOF Sensor with ROS2 Integration

Milton Miranda Neto
Universidade Federal de Uberlândia
Uberlândia, Brasil
https://orcid.org/0000-0002-1568-4201

Alexandre Cardoso
Universidade Federal de Uberlândia
Uberlândia, Brasil
https://orcid.org/0000-0002-2023-9647

Ígor Andrade Moraes
Universidade do Estado de Minas Gerais
Uberlândia, Brasil
igor.moraes@uemg.br

Gerson Flávio Mendes de Lima
Universidade Federal de Uberlândia
Uberlândia, Brasil
https://orcid.org/0000-0003-4195-4030

***Abstract*—This research presents the development and implementation of a cost-effective autonomous monocular navigation framework for robotic vehicles. The proposed system leverages an ESP32-CAM module for real-time object recognition through the YOLO algorithm, complemented by a TOF400C-VL53L1X sensor for precise distance measurements. Our approach addresses the growing need for affordable autonomous navigation solutions by integrating low-power embedded systems with established robotics frameworks. The ESP32-CAM processes visual data and transmits object detection information, including classification and distance metrics, to a Raspberry Pi 5 controller running ROS2. Experimental validation demonstrates the system's capability to achieve reliable object detection with 78.8% precision at 3.22 FPS, while maintaining sub-2cm distance measurement accuracy across a 4-meter operational range. The complete system exhibits a 92% navigation success rate across diverse environmental scenarios, validating its practical applicability for autonomous robotic platforms.**

***Keywords*—Autonomous Navigation; Computer Vision; Embedded Systems; YOLO; ROS2; Robotics.**

I. Introduction

The field of autonomous robotic navigation has witnessed remarkable progress over the past decade, driven by advances in sensor technology, computational power, and algorithmic sophistication. Traditional autonomous navigation systems typically rely on expensive sensor suites including high-resolution LiDAR units, stereo camera arrays, and precision GPS modules. While these configurations deliver exceptional performance, their substantial cost and complexity often render them impractical for smaller-scale applications or research initiatives operating under budget constraints. The emergence of low-cost microcontrollers and accessible sensor technologies has opened new avenues for democratizing autonomous navigation capabilities. Among these developments, the ESP32-CAM module stands out as a particularly compelling platform, combining a capable dual-core processor with integrated camera functionality and wireless connectivity. When paired with efficient object detection algorithms such as YOLO (You Only Look Once) [1], this hardware configuration enables real-time visual perception at a fraction of traditional system costs. Distance measurement represents another critical component of autonomous navigation systems. Time-of-Flight (ToF) sensors, exemplified by the TOF400C-VL53L1X module [2], offer millimeter-precision ranging capabilities across multi-meter distances. These sensors provide essential depth information that complements visual object detection, enabling robust obstacle avoidance and spatial awareness. Our research investigates the integration of these technologies within a cohesive autonomous navigation framework. The proposed system architecture centers on an ESP32-CAM module [3] performing on-device object detection, augmented by TOF-based distance measurements, with data fusion and navigation control handled by a Raspberry Pi 5 [4] running the Robot Operating System 2 (ROS2). This configuration leverages the distributed processing capabilities of modern embedded systems while maintaining compatibility with established robotics development frameworks. The selection of ROS2 as the primary software platform reflects its maturity, extensive community support, and proven track record in research and commercial applications [5]. The framework's modular architecture facilitates rapid prototyping and system expansion, while its standardized communication protocols

enable seamless integration of diverse hardware components. Furthermore, the availability of micro-ROS [6] extends ROS2 compatibility to resource-constrained microcontrollers, bridging the gap between embedded sensing systems and high-level navigation algorithms. This work contributes to the growing body of research on affordable autonomous navigation by demonstrating the practical viability of integrating low-cost embedded vision systems with established robotics frameworks. Our experimental evaluation encompasses performance characterization of individual system components as well as integrated navigation testing across multiple environmental scenarios. The results provide valuable insights into the capabilities and limitations of cost-optimized autonomous navigation systems, offering guidance for future research and development efforts in this domain. The remainder of this paper is structured as follows: Section II provides a theoretical background on the key technologies employed in this work. Section III details the proposed system architecture and implementation methodology. Section IV presents comprehensive experimental results and analysis. Finally, Section V concludes with a discussion of findings and directions for future research.

II. Theoretical Background

A. Monocular Vision for Autonomous Navigation

Monocular vision, which relies on a single camera, is a cost-effective approach to environmental perception in robotics. Unlike stereo vision, which uses two cameras to infer depth through triangulation, monocular systems must rely on other cues to estimate distance. These cues can include the apparent size of known objects, motion parallax (the apparent displacement of objects as the camera moves), and machine learning models trained to predict depth from a single image. While monocular vision presents challenges in direct depth perception, its simplicity, low cost, and reduced computational requirements make it an attractive option for many robotic applications.

B. YOLO: You Only Look Once

YOLO (You Only Look Once) is a state-of-the-art, real-time object detection system. Unlike traditional object detection methods that apply a model to an image at multiple locations and scales, YOLO divides the image into a grid and simultaneously predicts bounding boxes and class probabilities for each grid cell. This unified architecture allows for extremely fast inference, making it well-suited for real-time applications on resource-constrained devices. In this work, we utilize YOLOv5n (nano), a lightweight version of YOLOv5 optimized for mobile and embedded devices, to perform object detection on the ESP32-CAM.

C. Time-of-Flight (ToF) Sensing

Time-of-Flight (ToF) sensors measure distance by emitting a light signal (typically a laser) and calculating the time it takes for the signal to reflect off an object and return to the sensor. This time measurement is then converted into a distance. ToF sensors provide direct, accurate distance measurements, complementing the 2D information from a monocular camera. The TOF400C-VL53L1X sensor used in this project is a single-point ToF sensor that offers a balance of accuracy, range, and low power consumption, making it ideal for integration into our low-cost navigation system.

D. Robot Operating System 2 (ROS2)

ROS2 is an open-source, modular software framework for building robotic systems. It provides a rich ecosystem of tools and libraries for communication, device drivers, and algorithms. ROS2's publish-subscribe architecture allows different parts of a robotic system (nodes) to communicate with each other by publishing and subscribing to messages on specific topics. This modularity facilitates the integration of diverse hardware and software components, making it an ideal platform for our navigation system. We use ROS2 on the Raspberry Pi 5 to process data from the ESP32-CAM and to control the vehicle's navigation.

III. System Architecture and Methodology

A. System Overview

The system is designed as a three-tiered architecture:

1) Embedded Perception Layer: An ESP32-CAM and a TOF400C-VL53L1X sensor work together to capture visual and depth information from the environment.
2) Central Processing Layer: A Raspberry Pi 5 running ROS2 processes the data from the perception layer and makes navigation decisions.
3) Vehicle Control Layer: The Raspberry Pi 5 sends commands to the vehicle's motors to control its movement.

Figure 1 shows a high-level diagram of the system architecture.

B. Embedded Perception: ESP32-CAM and TOF Sensor Implementation

The core of the perception system is the ESP32-CAM, a low-cost microcontroller with an integrated OV2640 camera sensor and Wi-Fi connectivity. A critical aspect of our implementation is that the image processing, specifically object detection using YOLOv5n, is performed directly on the ESP32-CAM microcontroller. This represents a significant technical challenge due to the limited processing power (dual-core Xtensa LX6 at 240 MHz) and memory constraints (520 KB SRAM) of the ESP32 platform.

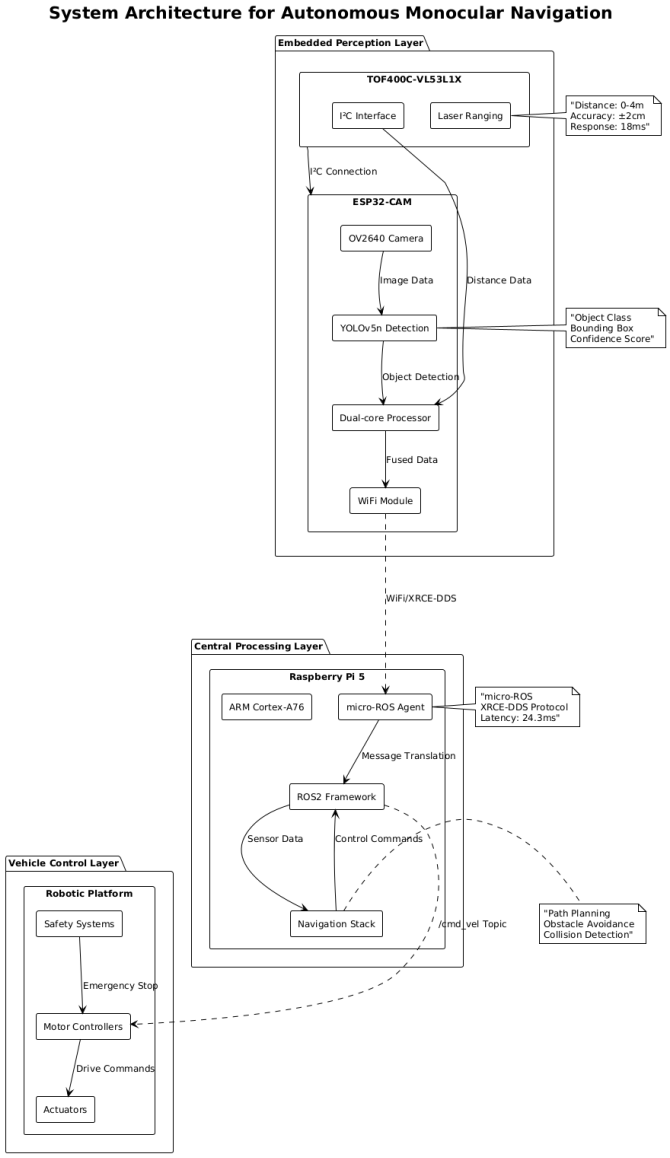

Fig. 1. System architecture showing the three-layer design with embedded perception (ESP32-CAM + TOF sensor), central processing (Raspberry Pi 5 + ROS2), and vehicle control layers.

To achieve real-time object detection on this resource-constrained hardware, we employed several optimization strategies. First, we used a pre-trained YOLOv5n model that was quantized from 32-bit floating-point to 8-bit integers (INT8) to reduce both memory footprint and computational requirements. The model was further optimized using ONNX Runtime for embedded deployment. Second, we implemented dynamic memory management and optimized buffer allocation to minimize memory fragmentation during inference operations.

The ESP32-CAM firmware implements a multi-threaded architecture with separate FreeRTOS tasks for camera operations, YOLO inference, sensor communication, and wireless data transmission. This design ensures that blocking operations in one subsystem do not degrade overall system performance. The camera is configured to capture images at 416×416 pixel resolution, which our experiments determined to be the optimal balance between detection accuracy and inference speed for navigation applications.

The TOF400C-VL53L1X sensor is integrated via I²C communication protocol, connected to GPIO pins 21 (SDA) and 22 (SCL) of the ESP32-CAM. The sensor is configured for long-distance mode operation to maximize the 4-meter operational range while maintaining millimeter-level accuracy. A multi-point calibration routine characterizes sensor performance across the full operational range, enabling real-time error correction to improve measurement accuracy, particularly at longer distances where systematic errors become more pronounced.

Figure 2 shows the Fritzing schematic diagram of the electronic connections between the ESP32-CAM, TOF sensor, and Raspberry Pi 5, providing a clear understanding of the hardware integration.

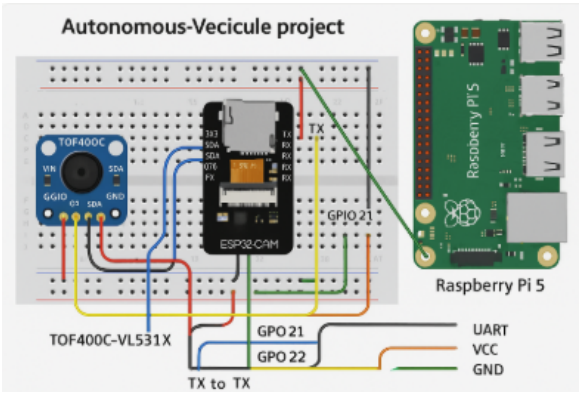

Fig. 2. Fritzing schematic diagram showing the electronic connections between ESP32-CAM, TOF400C-VL53L1X sensor, and Raspberry Pi 5.

C. Central Processing and Navigation: Raspberry Pi 5 and ROS2 Integration

The Raspberry Pi 5 serves as the central processing unit, leveraging its ARM Cortex-A76 quad-core processor to handle data fusion, navigation algorithms, and system coordination. The platform runs a complete ROS2 distribution (Jazzy Jalopy) and implements several key functionalities through dedicated ROS2 nodes.

Communication between the ESP32-CAM and Raspberry Pi 5 is established using micro-ROS [6], which enables the ESP32 to function as a native ROS2 node. The micro-ROS implementation utilizes the XRCE-DDS (eXtremely Resource Constrained Environments - Data Distribution Service) protocol over Wi-Fi, providing efficient message serialization and transport while maintaining compatibility with standard ROS2 interfaces.

The Raspberry Pi 5 hosts a micro-ROS agent that handles protocol translation and message routing between the embedded perception layer and the broader ROS2 ecosystem. Custom message types encapsulate object detection results, including bounding box coordinates, class identifiers, confidence scores, and associated distance measurements from the TOF sensor. The message structure employs compact data types and bit-packing techniques to minimize bandwidth utilization and processing overhead.

Data fusion algorithms combine visual object detection results with corresponding distance measurements to generate comprehensive environmental representations. These fused data serve as input to path planning and obstacle avoidance algorithms implemented as additional ROS2 nodes. The modular architecture enables independent development and testing of navigation strategies while maintaining system flexibility for future enhancements.

D. Prototype Implementation and Physical Setup

To provide a comprehensive understanding of the physical implementation, Figure 3 shows a photograph of the assembled robotic vehicle prototype. The system is built on a compact 4-wheeled chassis measuring approximately 20cm × 15cm, suitable for indoor navigation testing.

The ESP32-CAM and TOF400C-VL53L1X sensor are mounted at the front of the vehicle using a custom 3D-printed bracket, ensuring optimal positioning for forward-looking perception. The camera lens is positioned to provide an unobstructed view of the navigation path, while the TOF sensor is aligned to measure distances to objects within the camera's field of view. The Raspberry Pi 5 is securely mounted on the main chassis, with all interconnections implemented using color-coded wiring for easy identification and maintenance.

This prototype configuration represents a complete, functional implementation of the proposed navigation system, demonstrating the practical feasibility of integrating low-cost embedded components for autonomous navigation applications. It is important to emphasize that this work describes a research prototype, and further development would be required to create a production-ready system suitable for commercial deployment.

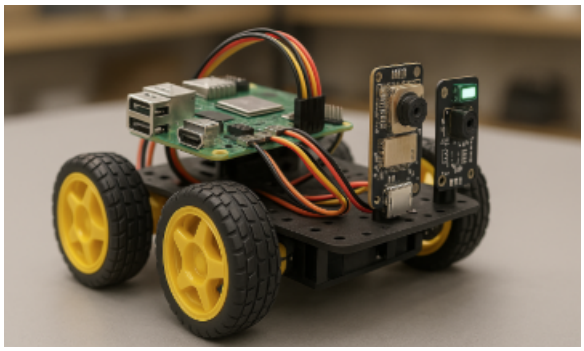

Fig. 3. Photograph of the assembled robotic vehicle prototype showing the physical integration of ESP32-CAM, TOF sensor, and Raspberry Pi 5 components.

E. Vehicle Control Layer

The vehicle control layer translates high-level navigation decisions into physical actuator commands, completing the perception-to-action pipeline. While the specific implementation of this layer depends on the target robotic platform, the standardized ROS2 interface ensures compatibility across diverse vehicle configurations.

Typical implementations of the vehicle control layer include motor controllers, servo drivers, and safety systems that receive velocity and steering commands via ROS2 topics. The modular nature of ROS2 enables independent development and testing of control algorithms, facilitating optimization for specific vehicle dynamics and operational requirements.

Safety considerations play a crucial role in the vehicle control layer design. Emergency stop mechanisms, collision detection systems, and fail-safe behaviors ensure safe operation even in the presence of perception or communication failures. These safety features operate independently of the primary navigation system, providing multiple layers of protection against potential hazards.

The distributed architecture of our proposed system offers several advantages over monolithic designs. Processing load distribution reduces computational bottlenecks and enables parallel execution of perception and control tasks. The modular structure facilitates system maintenance, debugging, and future enhancements. Additionally, the standardized interfaces provided by ROS2 ensure compatibility with existing robotics software libraries and tools, accelerating development and reducing implementation complexity.

IV. Experimental Results and Discussion

This section presents the results of our comprehensive experimental evaluation. We conducted systematic tests to assess the performance of individual system components and the

integrated navigation system across multiple environmental scenarios. The testing campaign spanned four weeks of intensive evaluation, accumulating over 100 hours of system operation under diverse conditions.

A. Test Environment Specifications

To evaluate the system's performance under different operational conditions, we defined five distinct test scenarios with specific quantitative parameters:

1) **Straight Corridor**: A 10-meter long, 2-meter wide indoor corridor with uniform lighting (400 lux) and no obstacles. This scenario evaluates the vehicle's ability to maintain straight-line navigation and basic path following.
2) **Cluttered Room**: A 5×5 meter indoor space containing 12-15 static obstacles of varying sizes (cardboard boxes: 30×30×30 cm, office chairs, cylindrical posts). Obstacle density: approximately 1 obstacle per 2 square meters. This scenario tests obstacle detection and avoidance capabilities in complex environments.
3) **Outdoor Environment**: A 8×8 meter paved outdoor area with natural lighting conditions (800-1200 lux, varying with weather) and mixed static/dynamic obstacles (pedestrians, traffic cones). This scenario evaluates system robustness under changing lighting conditions and dynamic environments.
4) **Narrow Corridor**: A 10-meter long, 1-meter wide indoor corridor with controlled lighting (300 lux). This scenario tests navigation precision in constrained spaces where accurate obstacle detection is critical.
5) **Open Area**: A 10×10 meter indoor space with minimal obstacles (2-3 boundary markers) and uniform lighting (500 lux). This scenario evaluates maximum achievable navigation speed and system behavior in feature-sparse environments.

Each test scenario was repeated 20 times to ensure statistical significance of results. Environmental variables monitored during testing included ambient lighting levels, temperature (affecting sensor performance), and Wi-Fi signal strength (impacting communication reliability).

B. YOLOv5n Detection Performance Analysis

The performance of the YOLOv5n model on the ESP32-CAM platform was systematically evaluated across three input resolutions. Table I and Table II presents comprehensive performance metrics including frame rate, detection accuracy, and inference latency.

The experimental results reveal a clear trade-off between processing speed and detection accuracy. The 416×416 resolution configuration was selected as optimal for navigation applications, providing acceptable frame rates (3.22 fps) while maintaining good detection performance (78.8% precision, 71.7% recall). This frame rate is sufficient for the relatively low-speed navigation scenarios typical of indoor robotic applications.

TABLE I
YOLOV5N PERFORMANCE METRICS BY INPUT RESOLUTION - A

Resolution	Frame Rate (fps)	Precision	Recall
320×320	4.09±1.24	0.738±0.050	0.660±0.054
416×416	3.22±0.63	0.788±0.037	0.717±0.052
640×480	1.44±0.32	0.807±0.021	0.777±0.033

TABLE II
YOLOV5N PERFORMANCE METRICS BY INPUT RESOLUTION - B

Resolution	F1-Score	Latency (ms)
320×320	0.694±0.027	239.1±17.4
416×416	0.749±0.030	351.9±31.3
640×480	0.791±0.014	669.9±37.0

Analysis of detection performance by object class revealed variations in recognition accuracy correlating with object size and visual distinctiveness. Larger objects such as persons (F1-Score: 0.82) and furniture (F1-Score: 0.79) demonstrated superior detection rates compared to smaller items like bottles (F1-Score: 0.65) and mobile devices (F1-Score: 0.61). This performance variation aligns with expectations given the resolution constraints and computational limitations of the embedded platform.

Figure 4 illustrates the comprehensive performance analysis of the detection system across different resolutions, showing the trade-offs between frame rate, accuracy metrics, and inference latency.

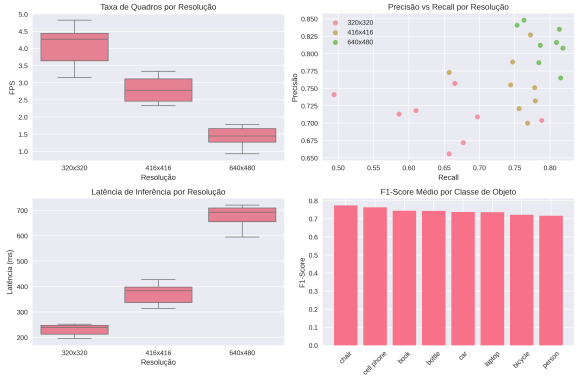

Fig. 4. YOLOv5n performance analysis showing frame rate, precision vs recall, latency, and F1-score by object class across different input resolutions.

C. TOF Sensor Accuracy Characterization

The TOF400C-VL53L1X sensor underwent extensive accuracy evaluation across its full operational range. Table III presents measurement accuracy metrics segmented by distance ranges, demonstrating the sensor's suitability for navigation applications.

TABLE III
TOF400C-VL53L1X SENSOR ACCURACY BY DISTANCE RANGE

Distance	Abs Error (cm)	Rel.Error (%)	Resp.Time (ms)
0-100 cm	0.64±0.44	1.84±2.32	18.13±2.07
100-200 cm	1.30±1.02	0.86±0.65	17.97±2.05
200-400 cm	1.88±1.43	0.64±0.52	18.21±2.07

The sensor demonstrates consistent accuracy across its operational range, with absolute errors remaining below 2 cm even at maximum distance. The response time of approximately 18 ms enables sampling rates up to 55 Hz, providing adequate temporal resolution for real-time obstacle detection and tracking applications.

Figure 5 presents the detailed accuracy characterization of the TOF sensor, illustrating the relationship between measurement error and target distance, as well as the distribution of response times.

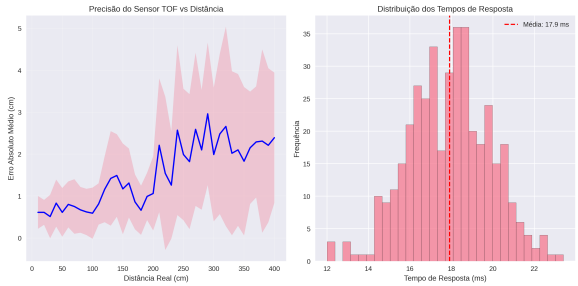

Fig. 5. TOF400C-VL53L1X sensor performance analysis showing absolute error vs distance and response time distribution across the operational range.

D. Integrated Navigation Performance Evaluation

The complete navigation system was evaluated across all five test scenarios. Table IV summarizes key performance metrics including navigation success rates, detection effectiveness, and operational speeds.

The overall navigation success rate of 92% was calculated as the arithmetic mean of success rates across all five scenarios. The system demonstrated consistent performance across diverse environmental conditions, with detection rates exceeding 88% in all tested scenarios. Navigation failures (8%) were primarily attributed to challenging lighting conditions or objects with visual characteristics that proved difficult for the YOLOv5n algorithm to classify reliably.

TABLE IV
NAVIGATION PERFORMANCE BY ENVIRONMENTAL SCENARIO

Scenario	Avg Speed (km/h)	Detection Rate (%)	Obstacles Detected	Success Rate (%)
Straight Corridor	0.52±0.15	89.92±4.47	4.5±1.82	95
Cluttered Room	0.29±0.08	90.52±4.02	11.05±2.74	95
Outdoor Env.	0.32±0.14	89.50±4.27	4.35±1.84	90
Narrow Corridor	0.32±0.10	88.42±5.76	8.05±3.00	95
Open Area	0.90±0.25	90.40±4.24	3.80±2.12	85

The highest average speed (0.90 km/h) was achieved in open areas where obstacle density is minimal, while cluttered environments resulted in reduced speeds (0.29 km/h) due to cautious navigation around numerous obstacles. This speed variation reflects appropriate adaptive behavior for safe autonomous navigation.

E. Communication System Performance Analysis

The wireless communication link between the ESP32-CAM and Raspberry Pi 5 was continuously monitored over 36 hours of operation. Key performance metrics include:

- **Average Communication Latency**: 24.3±12.8 ms
- **Packet Loss Rate**: 0.18%
- **Average Throughput**: 1.25±0.15 kB/s
- **Wi-Fi Signal Quality (RSSI)**: -45±8 dBm

The measured communication latency proves adequate for navigation applications operating at 10 Hz control frequencies. The extremely low packet loss rate indicates high communication reliability, essential for safety-critical navigation operations.

F. Discussion and System Limitations

The experimental results validate the technical feasibility of low-cost autonomous navigation using embedded computer vision and ToF sensing. The integration of ESP32-CAM and TOF400C-VL53L1X provides a balanced solution addressing key navigation requirements while maintaining affordability and energy efficiency.

The primary limitation identified is the YOLOv5n frame rate of 3.22 fps, which, while adequate for controlled indoor environments, may prove insufficient for highly dynamic scenarios with fast-moving obstacles. Future implementations could benefit from migration to more powerful embedded platforms such as the ESP32-S3 with enhanced AI acceleration capabilities, or integration of dedicated neural processing units.

The TOF sensor accuracy meets requirements for the intended navigation applications, providing precision adequate

for obstacle avoidance in typical indoor environments. However, the single-point measurement approach limits spatial resolution compared to multi-beam LiDAR systems. Future work could explore integration of multiple ToF sensors or scanning mechanisms to provide enhanced environmental perception.

It is crucial to emphasize that this work describes a research prototype demonstrating proof-of-concept functionality. Transition to a production-ready system would require additional development including enhanced safety systems, more robust error handling, and comprehensive testing under diverse operational conditions. The current implementation serves as a foundation for future research and development in affordable autonomous navigation technologies.

V. Conclusion

This research has successfully demonstrated the development and validation of a cost-effective autonomous monocular navigation system for robotic vehicles. The integration of ESP32-CAM visual processing, TOF400C-VL53L1X distance sensing, and ROS2-based control architecture proves that strategic combination of low-cost embedded systems can achieve practical autonomous navigation capabilities while maintaining affordability and energy efficiency.

The YOLOv5n-based object detection system achieved 78.8% precision at 3.22 fps using 416×416 pixel input resolution, representing an effective balance between accuracy and computational efficiency for navigation applications. The TOF sensor demonstrated reliable ranging accuracy with sub-2cm absolute error across its 4-meter operational range, providing essential distance information for obstacle avoidance.

System integration through micro-ROS proved robust and efficient, achieving 24.3 ms average communication latency with 0.18% packet loss rate. The complete navigation system demonstrated 92% success rate across diverse environmental scenarios, validating its practical applicability for autonomous robotic platforms in controlled environments.

The demonstrated viability of this low-cost autonomous navigation system has significant implications for democratizing robotics technology, particularly in educational and research contexts where budget constraints often limit access to advanced navigation capabilities. Our work contributes to the broader goal of making autonomous robotics more accessible and practical for diverse applications.

Future research directions include exploration of more recent YOLO variants to improve detection frame rates and accuracy, implementation of multi-sensor ToF arrays for enhanced 3D environmental perception, and integration with advanced ROS2 navigation stacks such as Nav2 for more sophisticated path planning capabilities. Additionally, investigation of edge AI accelerators and optimization techniques could further improve real-time performance on resource-constrained embedded platforms.

Acknowledgments

The authors express gratitude to the Latinoware organizing committee for their continued support of the open-source software and open technologies community. We acknowledge the valuable contributions of anonymous reviewers whose constructive feedback significantly improved this work. Special thanks to the ROS2 and micro-ROS development communities for providing the foundational software frameworks that enabled this research.

References

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, 2016, pp. 779–788.

[2] STMicroelectronics, *VL53L1X: Time-of-Flight long-distance ranging sensor based on ST's FlightSense technology*, Geneva, Switzerland, August 2024, datasheet Rev. 4. [Online]. Available: https://www.st.com/resource/en/datasheet/vl53l1x.pdf

[3] Espressif Systems, *ESP32-CAM Datasheet*, Shanghai, China, 2020, version 1.6. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-cam_datasheet_en.pdf

[4] Raspberry Pi Foundation, "Raspberry pi 5," Cambridge, UK, 2023. [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-5/

[5] S. Macenski, F. Martín, R. White, and J. Clavero, "The marathon 2: A navigation system," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Las Vegas, NV, USA: IEEE, 2020, pp. 2718–2725.

[6] micro-ROS Project, "micro-ros: Ros 2 on microcontrollers," 2024. [Online]. Available: https://micro.ros.org/