A Distributed ESP32-ROS Architecture for Real-Time EKF-SLAM on Low-Cost Mobile Robots

Soumyajeet Mahapatra
Dept. of Electrical Engineering
National Institute of Technology, Rourkela
Rourkela, India
smvssut17@gmail.com

Prof. Asim Kumar Naskar
Dept. of Electrical Engineering
National Institute of Technology, Rourkela
Rourkela, India
naskara@nitrkl.ac.in

***Abstract*—This paper presents a distributed computational architecture enabling real-time EKF-based state estimation on low-cost, resource-constrained mobile robots. Our framework strategically partitions tasks: an ESP32 microcontroller manages high-frequency, real-time odometry and LiDAR data acquisition, while a host computer running the Robot Operating System (ROS) executes the computationally demanding EKF-SLAM algorithm. This hybrid, low-cost system, which leverages the ESP32's built-in Wi-Fi for communication, is experimentally validated to produce accurate and stable state vector estimates $[x, y, \theta]^T$. This architecture provides a practical and scalable blueprint for implementing advanced estimation and control algorithms on inexpensive robotic hardware.**

***Index Terms*—State Estimation, EKF, SLAM, Resource-Constrained Systems, ESP32, ROS, Mobile Robotics, Control Systems**

I. Introduction

Robust autonomous navigation requires solving the Simultaneous Localization and Mapping (SLAM) problem. The Extended Kalman Filter (EKF) is a powerful probabilistic tool for SLAM, but its computational complexity, which grows quadratically $\mathcal{O}(n^2)$ with the number of landmarks (n), makes it unfeasible for low-cost microcontrollers [1].

Hardware such as the ESP32 is ideal for inexpensive robots due to integrated Wi-Fi and I/O ports, but it lacks the resources for the large operations required by EKF-SLAM. This work addresses this challenge by proposing a distributed, hybrid architecture that partitions the computational load. We delegate all real-time sensor processing to the resource-constrained ESP32 and offload the high-level, computationally intensive EKF estimation to a host PC running the Robot Operating System (ROS) [2].

Our contribution is the validation of this practical framework, which fuses data from low-cost sensors such as single-channel encoders and 2D LiDAR to achieve high-fidelity localization, demonstrating a path to advanced autonomy on inexpensive hardware.

II. Distributed System Architecture

Our architecture, shown in Fig. 1, decouples hard real-time sensing from complex, non-real-time estimation.

A. Real-Time Sensing Client (ESP32)

A single ESP32-WROOM-32 microcontroller manages all hardware interfacing. Its firmware, written in C++ (Arduino framework), is responsible for:

- **Motor Control & Odometry:** Reading low-cost MOC7811 single-channel optical encoders via hardware interrupts to compute linear and angular velocity.
- **LiDAR Interfacing:** Buffering raw data from a YDLIDAR X2 scanner via UART, without any filtering or processing.
- **Communication:** Using the `ros-serial` TCP protocol over Wi-Fi, the ESP32 connects to the ROS master and publishes sensor data directly as standard ROS messages: `nav_msgs/Odometry` (from encoders) and `sensor_msgs/LaserScan` (from LiDAR).

This design dedicates the ESP32 to time-critical tasks, preventing the EKF's computational load from interfering with sensor data acquisition.

B. Computational Server (ROS Host)

A host PC (Ubuntu 20.04, ROS Noetic) subscribes to the `/odom` and `/scan` topics published by the ESP32. The host's responsibilities are:

- **Landmark Detection:** Processing the raw `/scan` data to perform clustering and circle-fitting (via least-squares to extract landmark positions.
- **EKF-SLAM Estimation:** Running the main EKF-SLAM node that performs all matrix operations for the state estimation [5].

This partitioning allows the computationally expensive algorithms to run on suitable hardware without compromising the real-time performance of the robot's low-level sensors [3].

III. EKF-SLAM Algorithm

The core logic is executed on the ROS host. It subscribes to the `/odom` topic (for the prediction) and `/scan` topic (for the update). Algorithm 1 outlines the recursive estimation process, which estimates the state vector μ (containing robot pose and landmark positions) and its covariance Σ.

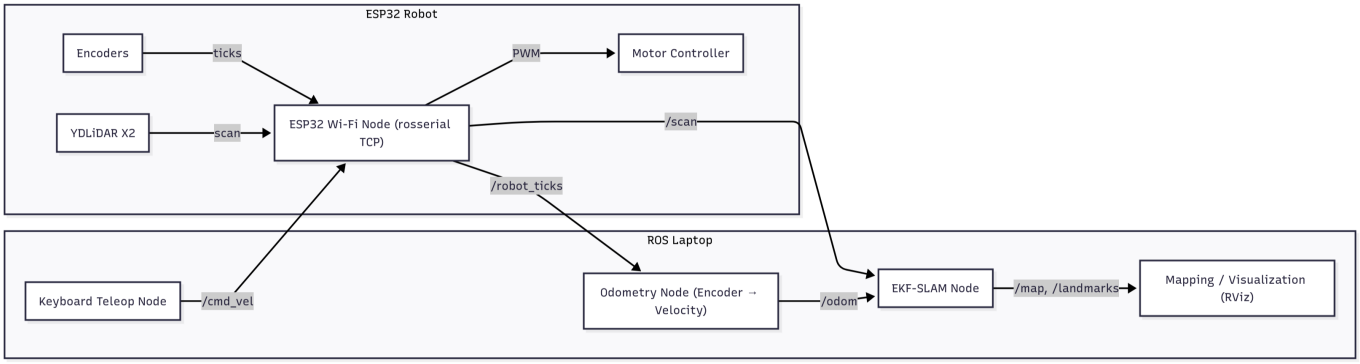

Fig. 1. System architecture for distributed EKF-SLAM, showing the clear partition of tasks between the ESP32 client and the ROS host, connected via Wi-Fi.

Algorithm 1 EKF-SLAM Step

function EKF_SLAM(μ_k, Σ_k, u_{k+1}, z_{k+1})
{— Prediction (on /odom) —}
$\overline{\mu}_{k+1} = g(\mu_k, u_{k+1})$ Predict state via motion model
$G_{k+1} = \nabla_\mu g(\mu_k, u_{k+1})$ Compute motion Jacobian
$\overline{\Sigma}_{k+1} = G_{k+1}\Sigma_k G_{k+1}^T + R_{k+1}$ Predict covariance
{— Update (on /scan) —}
for each landmark j in observation z_{k+1} **do**
  $\hat{z}^j = h(\overline{\mu}_{k+1})$ Expected measurement
  $H_{k+1}^j = \nabla_\mu h(\overline{\mu}_{k+1})$ Measurement Jacobian
  $S = H_{k+1}^j \overline{\Sigma}_{k+1} (H_{k+1}^j)^T + Q_{k+1}$ Innovation cov
  $K = \overline{\Sigma}_{k+1} (H_{k+1}^j)^T S^{-1}$ Kalman Gain
  {Update state and covariance}
  $\mu_{k+1} = \overline{\mu}_{k+1} + K(z_{k+1}^j - \hat{z}^j)$
  $\Sigma_{k+1} = (I - K H_{k+1}^j)\overline{\Sigma}_{k+1}$
  {Pass updates to next iteration in loop}
  $\overline{\mu}_{k+1} = \mu_{k+1}$
  $\overline{\Sigma}_{k+1} = \Sigma_{k+1}$
end for
return μ_{k+1}, Σ_{k+1}
end function

IV. Experimental Validation

We validated the system in a 10m x 8m lab environment with cylindrical landmarks. The robot was driven manually with the help of a custom keyboard teleop ROS node, building the map in real-time. The actual arena is shown in Fig. 2.

A. Computational Performance

The primary validation of our architecture is the computational load on the microcontroller.

- **ESP32 CPU Load:** Monitored throughout the experiment, the ESP32's firmware (handling encoders, LiDAR data, and Wi-Fi) consumed an average of only 22% of the total CPU time.

This result confirms that the $\mathcal{O}(n^2)$ EKF complexity is fully offloaded, leaving the ESP32 with $\approx$ 78% computational headroom for more advanced, low-level control tasks. On the host PC, the EKF update time for a 4-landmark map was 18.5 ms, well within the 200 ms budget allowed by the 5 Hz LiDAR, confirming real-time feasibility.

Fig. 2. Experimental Arena

B. Localization and Mapping Accuracy

The qualitative mapping result is shown in Fig. 3. The system successfully generated a map that is structurally consistent with the real environment.

- **Map Accuracy:** The Root Mean Square Error (RMSE) of the estimated landmark positions, compared against ground-truth locations, was 3 cm.

This high-fidelity result, achieved with very low-cost and noisy sensors (single-channel encoders), validates the robustness of the distributed sensor fusion approach.

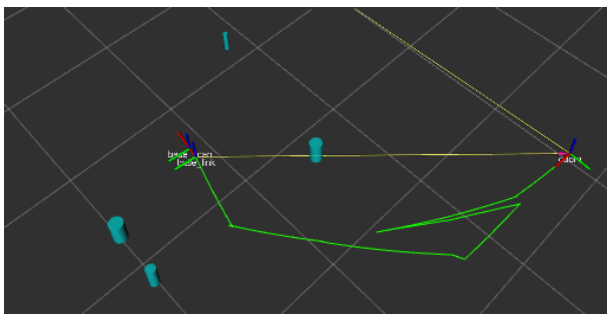

Fig. 3. Final map and robot trajectory visualized in RViz.

V. Conclusion and Future Work

This paper presented and validated a distributed architecture for EKF-SLAM on low-cost, resource-constrained robots. By partitioning tasks between an ESP32 for real-time sensing and a ROS host for computation, we demonstrated a system that is both robust and computationally feasible. Our implementation achieved a 3 cm landmark RMSE while imposing only a 22% CPU load on the microcontroller, providing a practical blueprint for enabling advanced autonomy on accessible hardware.

Future work will leverage this framework to implement and compare other SLAM algorithms, such as Graph-Based SLAM and FastSLAM (a Rao-Blackwellized Particle Filter), to analyze their performance and scalability on the same low-cost platform.

References

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2005.

[2] A. L. M. V. M. da Silva, et al., "Integration of ESP32 with ROS for mobile robot applications," in *2021 IEEE International Conference on Automation/XXIV C-MAC*, 2021, pp. 1-6.

[3] R. G. R. de Oliveira, et al., "A Distributed Architecture for Autonomous Mobile Robots Using ROS and ESP32," in *2019 IEEE Latin American Robotics Symposium (LARS)*, 2019, pp. 1-6.

[4] J. J. Leonard and H. F. Durrant-Whyte, "Simultaneous map building and localization for an autonomous mobile robot," in *Proc. IEEE/RSJ Int. Workshop Intell. Robots Syst.*, 1991, pp. 1442–1447.

[5] S. Thrun, "A probabilistic approach to concurrent mapping and localization for mobile robots," *Machine Learning*, vol. 31, no. 1, pp. 29–53, 1998.