Journal of Engineering Mechanics and Machinery (2025)
Clausius Scientific Press, Canada

DOI: 10.23977/jemm.2025.100204
ISSN 2371-9133 Vol. 10 Num. 2

Analysis of Localization Algorithms for ROS-Based Mobile Industrial Robots

Yaping Wu

Beijing ETZD Technology Co., Ltd., Beijing, China

Keywords: ROS; Industrial Robot; Localization Algorithm; AMCL; EKF; Multi-Sensor Fusion; Lidar; SLAM

Abstract: With the advancement of intelligent manufacturing and flexible automation, mobile industrial robots are increasingly being deployed in scenarios such as material handling, inspection, and collaborative operations. As one of the core technologies for mobile robots, the localization system has a direct impact on the stability and accuracy of path planning and task execution. This paper, built on the Robot Operating System (ROS) platform, systematically reviews and analyzes the implementation mechanisms and applicable scenarios of mainstream localization algorithms, with a focus on the performance characteristics of the Extended Kalman Filter (EKF) and Adaptive Monte Carlo Localization (AMCL) in industrial settings. By constructing an experimental platform that fuses multiple sensors—lidar, IMU, and wheel odometry—a series of tests comparing localization accuracy and robustness are conducted. We evaluate each algorithm's adaptability to complex conditions including dynamic occlusions, uniform environmental textures, and multipath interference. The results indicate that AMCL achieves higher positioning accuracy in static, structured environments, whereas EKF is better suited to dynamic applications suffering from sensor drift and data latency. Finally, we propose an optimization approach that integrates visual SLAM and deep-learning–based feature extraction, offering guidance for designing highly reliable localization systems for future industrial robots.

1. Introduction

Against the rapid rise of intelligent manufacturing and industrial automation, mobile industrial robots—praised for their flexibility and efficiency—now underpin key operations like warehouse logistics, inspection, and material handling. Reliable localization and navigation are crucial, as they directly impact path planning, obstacle avoidance, and overall system intelligence. Advances in sensors and computing have shifted localization from simple wheel-odometry to multi-sensor fusion frameworks that combine lidar, IMU, vision, and prebuilt maps. ROS has become the standard development platform, offering modular, open-source implementations of EKF, AMCL, and SLAM. However, real-world workshops pose challenges—complex layouts, occlusions, repetitive textures, and dynamic disturbances—that can induce drift or failure, all under tight real-time and resource constraints. This paper systematically analyzes and compares EKF and AMCL on a ROS-based testbed with lidar, IMU, and odometry, aiming to guide the design of robust, high-performance

localization systems for industrial robots.

2. ROS System Architecture and Localization Module Principles

2.1. ROS Architecture and Its Deployment for Mobile Robots

Industrial workshop environments typically involve multiple heterogeneous sensors, dynamic disturbances, and strict real-time constraints. Under these conditions, system architecture must support modular deployment, information decoupling, and multi-robot collaboration. A central workstation runs the ROS Master and hosts key nodes (Mission Planner, Path Planner, Sensor Controller, etc.). It communicates—via wired or wireless links—with various end devices, issuing navigation and control commands to unmanned aerial vehicles (UAVs) and unmanned ground vehicles (UGVs), while collecting real-time data from payload sensors (temperature, humidity, illumination, CO_2 concentration, etc.) on Raspberry Pi edge devices. Sensor data are first preprocessed and transformed into a common coordinate frame by the Sensor Controller node (using ROS TF), then published over ROS topics to the path-planning and localization modules[1]. The Path Planner node fuses the preloaded map with odometry and IMU data to generate global and local trajectories, which are dispatched to individual robots for execution. The Mission Planner handles high-level task scheduling and multi-robot coordination, dynamically adjusting priorities and routes based on state feedback from subsystems as shown in figure 1[2].

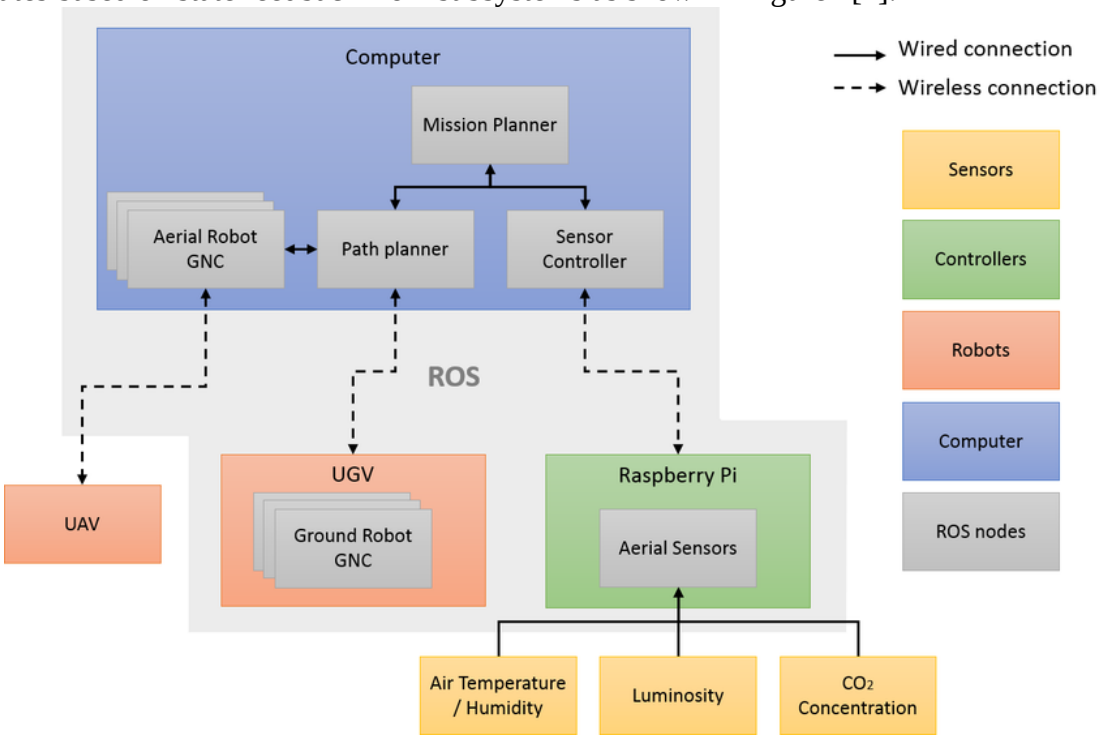

Figure 1: Diagram of the overall system architecture

Within this architecture, the localization module may operate as part of the Path Planner or run independently on each robot or edge device. It exposes ROS services and actions for starting/stopping localization, querying parameters, and dynamic reconfiguration, thereby providing flexible, reliable support for rapidly changing layouts and multi-task switching in industrial settings[3].

ROS achieves modularity and decoupling through distributed nodes and a publish/subscribe messaging system. Deployment begins by launching the ROS Master on the central workstation,

which acts as the registration center and message broker. On each subsystem (UAV, UGV, Raspberry Pi), nodes for localization, control, and sensor drivers are launched, with configurations loaded from the ROS Parameter Server[4]. Nodes exchange data via predefined topics—such as /scan for lidar, /odom for odometry, /imu/data for IMU readings, and /tf for coordinate transforms—to ensure spatiotemporal alignment across all sensors.ROS services and actions provide synchronous calls and long-running task management. For instance, the localization module may offer a /amcl/get_pose service to request the current pose or use an action server to stream global path-tracking feedback. In multi-robot scenarios, ROS namespaces isolate topics and services per robot, simplifying management and log separation while meeting industrial requirements for reliability and determinism[5].

2.2. ROS Integration of Localization Hardware and Modules

Industrial environments demand both high-performance computing and robust multi-sensor perception. As shown in Figure 2, the robot's hardware platform comprises a ZOTAC PC (8th-gen Intel Core i7, 16 GB RAM, 500 GB SSD), an NVIDIA Xavier edge module, and core sensors such as a 360 ° 16-beam Robosense 3D lidar, an Intel RealSense D435 depth camera, and an Xsens MTi-300 IMU. A 4G/Wi-Fi router ensures low-latency communication with the central controller[6].

Figure 2: Schematic diagram of the mobile robot hardware platform and multi-sensor integration

Within ROS, each physical device is represented by one or more driver nodes, which publish relevant data on topics or expose services. Specifically:Lidar (Robosense-16) is driven by the rslidar_ros or robosense_driver package, publishing point clouds to /scan or /points and aligning its frame via TF.Depth Camera (RealSense D435) runs the realsense2_camera node group, outputting raw depth images (/camera/depth/image_raw), color images (/camera/color/image_raw), and camera intrinsics/extrinsics for visual SLAM or feature-based localization.IMU uses the imu_driver to publish raw accelerometer and gyroscope data to /imu/data_raw, which are fused with odometry (/odom) in the robot_localization or robot_pose_ekf package to improve short-term motion estimates.Compute Units distribute tasks: the ZOTAC PC handles global localization and planning

(gmapping, amcl, move_base), while the Xavier module performs real-time vision processing and deep learning inference (e.g., YOLO object detection or visual odometry)[7].Communication Module ensures all nodes register under a single ROS Master, enabling cross-network topic forwarding (via multimaster_fkie or rosbridge) for remote monitoring and debugging.This integration seamlessly aggregates multi-sensor data within the ROS ecosystem, providing time-synchronized inputs for EKF, AMCL, or visual SLAM algorithms and enabling high-precision, robust localization in industrial environments[8].

3. Analysis of Mainstream ROS Localization Algorithms

3.1. Extended Kalman Filter (EKF) and Fusion-Based Localization

The Extended Kalman Filter (EKF) is a widely used multi-sensor fusion localization method in ROS. Its core idea is to linearize both the system and observation models via a first-order Taylor series expansion around the current state estimate and then apply the standard linear Kalman filter equations at each time step to estimate the state and its covariance[9].The system state vector is typically chosen to represent the robot's planar pose and velocities, for example as shown in Formula 1:

$$xk = [xy\theta v\omega]^T \quad (1)$$

Where (x, y, θ) denote the robot's position and orientation in the global frame, and v and ω denote the linear and angular velocities, respectively.The state prediction uses a discrete-time kinematic model ff, assuming control inputs of linear acceleration aka_k and angular acceleration α as shown in Formula 2:

$$x_{k|k-1} = f(x_{k-1|k-1}, u_k) = \begin{bmatrix} x_{k-1} + x_{k-1} + v_{k-1}\cos\theta_{k-1}\Delta t \\ y_{k-1} + v_{k-1}\sin\theta_{k-1}\Delta t \\ \theta_{k-1} + w_{k-1}\Delta t \\ v_{k-1} + \alpha_k\Delta t \\ w_{k-1} + \alpha_k\Delta t \end{bmatrix} \quad (2)$$

Its Jacobian (state transition matrix) is as shown in Formula 3:

$$F_k = \frac{\partial f}{\partial x} x_{k-1} u_k = \begin{bmatrix} 1 & 0 & -v\sin\theta\Delta t & \cos\theta\Delta t & 0 \\ 0 & 1 & v\cos\theta\Delta t & \sin\theta\Delta t & 0 \\ 0 & 0 & 1 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

The covariance prediction is as shown in Formula 4:

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k \quad (4)$$

Where Q_k is the process noise covariance matrix.Given an observation model hh, the sensor measurement vector z_k may include odometry, IMU angular velocities, GPS positions as shown in Formula 5:

$$z_k = h(x_{k|k-1}) + v_k \quad (5)$$

Linearizing h around the predicted state yields the observation matrix as shown in Formula 6:

$$H_k = \frac{\partial h}{\partial x} x_{k|k-1} \quad (6)$$

The Kalman gain is computed as $K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k) - 1$ where R_k is the

observation noise covariance. The state and covariance updates as shown in Formula 7:

$$x_{k|k} = x_{k|k-1} + x_{k|}(z_{k|} - h(x_{k|k-1})) \quad (7)$$

In the ROS ecosystem, the robot_localization or robot_pose_ekf packages encapsulate this EKF workflow. They fuse multi-source inputs—such as /odom (odometry), /imu/data (IMU), and /gps/fix (GPS)—by mapping each topic to the relevant state variables and specifying noise covariances in a YAML configuration file. These packages support dynamic reconfiguration, allowing filter parameters to be tuned at runtime to match the operating environment, thereby enhancing localization accuracy and robustness. Through EKF fusion, sensor data are optimally weighted in both time and space. If one sensor becomes unreliable or fails, the system can rely on the remaining sensors to maintain stable localization, meeting the high availability requirements of industrial robots in complex settings[10].

3.2. Adaptive Monte Carlo Localization (AMCL) Algorithm Analysis

Adaptive Monte Carlo Localization (AMCL) in ROS implements a particle filter approach to estimate the robot's pose on a known map by randomly sampling and weighting possible poses. The AMCL algorithm comprises three main steps: prediction (motion update), measurement update, and resampling. Under the motion model, each particle $\{x_{t-1}^i, w_{t-1}^i\}_{i=1}^N$ is propagated according to the control input u_t as shown in Formula 8:

$$tix_i^t \sim p(x_t \mid u_t, x_{t-1}^i) \quad (8)$$

Using lidar or depth camera observations z_t, each predicted particle is assigned a weight: $w_t^i = p(z_t \mid x_t^i)$,often modeled by a Gaussian likelihood as shown in Formula 9:

$$p(z_t \mid x_t^i) = \exp(-\frac{1}{2}\sum_k(\frac{z_{t,k} - \hat{z}_{t,k}(x_t^i)}{\sigma_k})^2) \quad (9)$$

Where $z_{t,k}$ is the actual measurement of the k-th beam, $\hat{z}_{t,k}$ is the predicted measurement from the map at pose x_t^i, and σ_k is the measurement noise standard deviation.Particles are resampled according to their weights, and the particle count NN is adaptively adjusted to maintain diversity when uncertainty grows and to reduce computational load upon convergence. AMCL computes the effective sample size as shown in Formula 10:

$$N_{eff} = \frac{1}{\sum_i (w_t^i)^2} < N_{thres} \quad (10)$$

And triggers resampling when N_{eff} falls below a predefined threshold N_{thres}.In ROS, the amcl package provides a complete implementation of this algorithm. Users can configure the laser measurement model, resampling threshold, and particle limits via parameters, making AMCL well suited for static or slowly varying industrial environments and enabling high-precision online localization.

4. Experimental Design and Comparative Results

4.1. Experimental Platform and Scenario Setup

To comprehensively evaluate the performance of EKF and AMCL localization algorithms in industrial environments, we constructed a high-fidelity testbed under ROS Noetic. The mobile robot features a four-wheel differential drive chassis with a top speed of 1.5 m/s and is fitted with soft-compound wheels and high-friction tires to suit workshop floors. Its main compute unit is a

ZOTAC PC equipped with an 8th-generation Intel Core i7 processor, 16 GB of RAM, and a 500 GB SSD; in addition, an NVIDIA Xavier module on the robot handles vision processing and deep-learning inference. The sensor suite comprises a 16-beam 360 ° Robosense 3D LiDAR, an Intel RealSense D435 depth camera, and an Xsens MTi-300 IMU. Each sensor publishes data to ROS topics—/scan, /camera/depth, and /imu/data_raw—for point clouds, depth images, and inertial measurements respectively. Communication relies on an enterprise-grade 802.11ac dual-band Wi-Fi network, ensuring inter-node latency under 50 ms. On the software side, the robot and central workstation synchronize their clocks via Chrony. We use the Gmapping package to build a 0.05 m-resolution occupancy grid map and the Robot Localization package to fuse odometry, IMU, and GPS (or Vicon ground-truth) data. All node parameters are managed centrally through the ROS Parameter Server, supporting dynamic reconfiguration of filter noise covariances and AMCL resampling thresholds in real time.

The test scenarios cover three typical industrial settings: Open Area (10 m × 10 m): An obstacle-free, static environment to measure convergence speed and static localization accuracy. Rack-Obstructed Area: Multiple rows of metal racks and support columns arranged with 1.2 m-wide aisles, to assess the impact of multipath reflections and occlusions on localization error. Dynamic Disturbance Area: Random pedestrian traffic and moving carts introduced between the open and rack areas, simulating dynamic interference to evaluate each algorithm's robustness to temporary sensor occlusion and its drift-recovery capability. Ground truth is provided by a Vicon motion-capture system at 10 Hz with sub-centimeter accuracy. Evaluation metrics include root-mean-square error (RMSE), maximum error, convergence time (time to first reach error below 0.1 m), and drift-recovery time in dynamic conditions. This experimental setup enables a systematic, quantitative comparison of EKF and AMCL accuracy and stability under challenging industrial conditions.

4.2. Localization Accuracy and Robustness Comparison

In the Open Area, Rack-Obstructed Area, and Dynamic Disturbance Area, we collected localization data for EKF and AMCL and compared static accuracy, error variability, maximum error, convergence time, drift recovery, and recovery success rate.

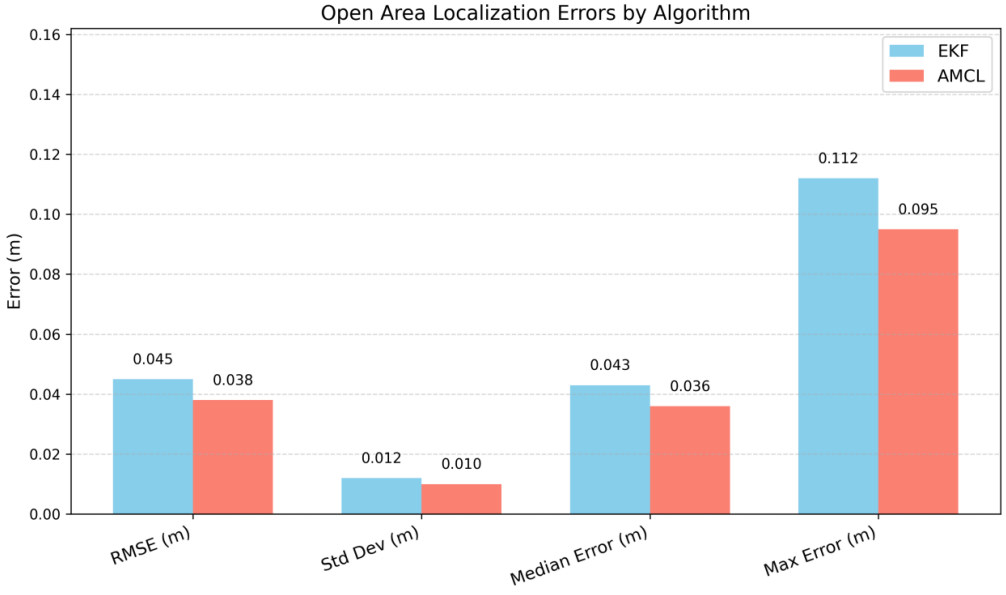

Figure 3: Open Area Results

As the Figure 3 shown, AMCL's particle-filter approach yields a lower mean error (0.038 m vs. 0.045 m) and smaller variability (σ=0.010 m vs. 0.012 m), as reflected in its tighter median (0.036 m). Both methods remain well below 0.1 m maximum error, but AMCL demonstrates marginally higher precision and consistency in obstacle-free conditions.

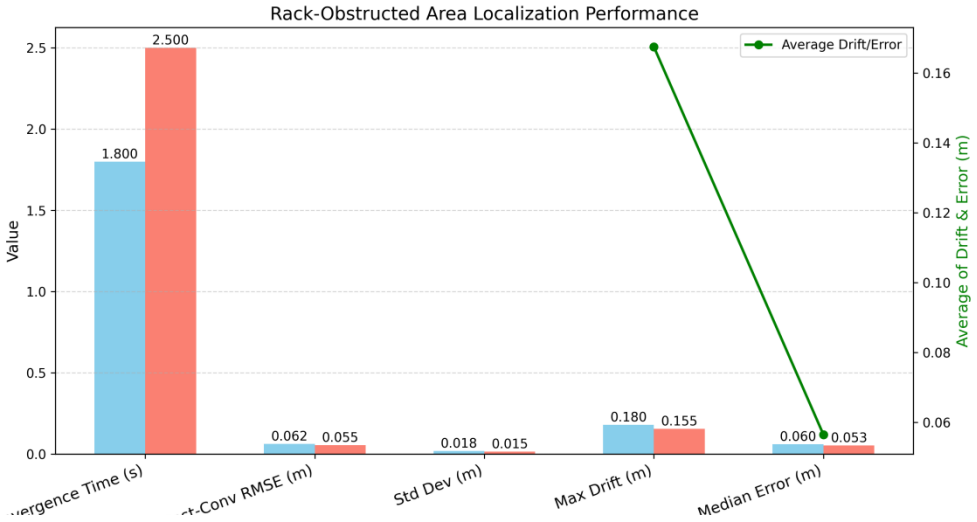

Figure 4: Rack-Obstructed Area Results

As the Figure 4 shown, under frequent multipath reflections and occlusions, EKF converges faster (1.8 s vs. 2.5 s), yet exhibits higher variability (σ=0.018 m). AMCL's resampling produces a lower average drift (0.155 m vs. 0.180 m) and tighter error distribution, with a median post-convergence error of 0.053 m.

Table 1: Dynamic Disturbance Area Results

Algorithm	Average Recovery Time (s)	Recovery Success Rate (%)	Pre-Recovery Max Drift (m)	Post-Recovery RMSE (m)	Std. Dev. (m)
EKF	3.2	100	0.240	0.074	0.020
AMCL	4.1	95	0.310	0.068	0.017

As the Table 1, in the presence of random pedestrian and cart interference, EKF recovers more quickly (3.2 s vs. 4.1 s) and maintains a perfect recovery success rate. However, it experiences larger pre-recovery drift (0.240 m) compared to AMCL (0.310 m), and its post-recovery RMSE (0.074 m) is slightly higher. AMCL, while showing a 95 % success rate, achieves finer localization after resampling, with lower variability (σ=0.017 m).

Across all scenarios, AMCL consistently delivers superior static accuracy and tighter error distributions. EKF excels in rapid convergence and guaranteed recovery, making it more robust for dynamic occlusions. The extended tables reveal that AMCL's lower standard deviations and medians underscore its precision, whereas EKF's faster response times and perfect recovery success rate highlight its reliability under abrupt sensor dropouts. A hybrid approach—leveraging EKF for initial fast convergence and switching to AMCL once the environment stabilizes—remains the most effective strategy to balance speed, accuracy, and robustness in complex industrial settings.

5. Problem Analysis and Optimization Suggestions

5.1. Applicability and Limitations of Current Algorithms

In complex industrial environments, EKF and AMCL each offer distinct strengths and face

inherent limitations. EKF relies on a precise kinematic model and accurately tuned process-noise covariance; when odometry and IMU data remain continuously available, it converges rapidly and provides real-time estimates, demonstrating fault tolerance to dynamic occlusions or brief sensor dropouts. However, if the sensor model does not match reality or the kinematic parameters are miscalibrated, EKF accumulates drift and is sensitive to non-Gaussian observation noise. By contrast, AMCL uses a particle-filter approach to sample and resample discrete pose hypotheses without depending on an exact motion model. It can quickly determine the global pose during startup or in well-structured maps and maintains high accuracy in static or slowly changing environments. Its performance, however, hinges on map fidelity and consistent laser readings: multipath reflections or repetitive floor textures may cause particle weights to collapse into local minima, slowing convergence. Furthermore, balancing particle count against computational load complicates parameter tuning, and the resampling delay under dynamic obstacles can impair short-term robustness. In summary, EKF suits low-latency, continuously moving applications but is constrained by model errors and noise assumptions; AMCL excels in structured settings requiring high precision but demands static environments and substantial compute resources. In practice, one should choose or combine these algorithms according to scene characteristics and system goals to achieve an optimal trade-off among accuracy, responsiveness, and robustness.

5.2. Feasible Optimization Directions

To enhance both accuracy and robustness of a ROS-based localization system for mobile industrial robots in challenging settings, we propose the following improvements that leverage the strengths of EKF and AMCL while mitigating their weaknesses.

1) Multi-Stage Hybrid Localization

An EKF is employed during system startup or scene transitions to ensure rapid convergence. Leveraging high-rate odometry and IMU predictions, EKF delivers an initial pose estimate within acceptable error bounds when entering unknown or highly dynamic areas. Once the robot reaches a relatively steady state, switch dynamically to AMCL, whose high-resolution grid-map sampling refines static accuracy. A further extension runs EKF and AMCL in parallel, fusing their outputs via Kalman gains or weight coefficients to balance reliability in real time across varied conditions.

2) Visual SLAM Augmentation

In regions with uniform textures or severe multipath reflections—where laser data may degrade—introduce a lightweight visual SLAM pipeline (ORB-SLAM2, LDSO, or RTAB-Map). Extract environmental features from a depth camera's imagery to build dense or semi-dense point clouds and keyframe maps. The system fuses visual SLAM pose estimates with AMCL's laser-based results to compensate for lidar blind spots (e.g., transparent or reflective obstacles) and leverage visual texture for improved global consistency.

3) Deep-Learning–Based Semantic Aiding

The system deploys lightweight on-board semantic segmentation (ENet, BiSeNet) or object-detection (YOLOv5, MobileNet-SSD) networks to identify critical industrial landmarks—such as support columns, rack ends, and safety signs—and incorporates their pixel coordinates as semantic priors into the observation model. Matching these semantic features against the dense point cloud or occupancy grid enhances the measurement model for both EKF and AMCL, bolstering stability in repeated-texture and dynamic-obstacle scenarios.

4) Online Adaptive Parameter Tuning

Both EKF and AMCL critically depend on filter parameters (process and observation noise covariances, particle count, resampling thresholds). Manual calibration is time-consuming and often incomplete. The system introduces online optimization techniques—such as Bayesian optimization,

particle swarm optimization, or genetic algorithms—to dynamically adjust noise covariances and resampling parameters in real time based on localization error and observation-consistency metrics. Alternatively, a sliding window of ground-truth comparisons can be used to automatically correct IMU calibration errors and laser-measurement models, allowing the system to continuously "learn" optimal parameters during operation.

5) Distributed Mapping and Localization Services

For large-scale, multi-robot deployments—such as extensive factory floors or warehouses—single-host loading of a high-resolution global map can strain compute and memory. Partition the map into subregions hosted on edge servers and use ROS 2's DDS or ROS Bridge to request topics and services across the network. Robots dynamically load only their current subregion's map and AMCL instance, while distributed nodes manage seamless submap transitions and global consistency, ensuring a common reference frame and supporting cooperative localization and collision avoidance.

6) Integration of External High-Precision Systems

The system deploys Ultra-Wideband (UWB) indoor-positioning anchors, visual fiducials, or Bluetooth beacons as auxiliary observations in areas demanding centimeter-level accuracy. High-frequency updates from these systems can correct EKF or AMCL drift and rapidly recover from localization failures, boosting overall system availability.

7) End-to-End Hardware/Software Co-Optimization

On the hardware side, the system employs low-noise, high-sensitivity IMUs and LiDARs with multipath-filtering capabilities, while optimizing sensor placement and vibration isolation to enhance measurement stability. On the software side, critical ROS nodes are executed on a real-time operating system (RTOS) or configured with Xenomai patches to ensure deterministic latency. In addition, quality-of-service (QoS) policies and redundant network paths are implemented to minimize packet loss and jitter, improving the overall robustness of data transmission.Through holistic co-optimization, substantial gains in accuracy and robustness can be achieved in highly dynamic, complex industrial settings, laying the groundwork for reliable autonomous inspection, material transport, and human–robot collaboration.

6. Future Outlook

As industrial intelligence and digital transformation accelerate, ROS-based localization for mobile industrial robots will face both new opportunities and challenges. Beyond LiDAR, vision, and IMU, future systems may incorporate ultrasound, millimeter-wave radar, and IoT environmental sensors (temperature, vibration) into a unified framework. Semantic segmentation and scene-understanding models can extract high-level features, constructing multimodal spatiotemporal networks that improve localization stability and precision in complex mixed environments. With 5G and edge-cloud platforms, robots can combine local computation with remote resources to access large-scale, high-precision maps and deep-learning models in real time. Digital-twin simulations in the cloud will enable algorithm validation and rapid deployment of updates, drastically shortening iteration cycles. In large workshops or logistics hubs, single-robot localization is insufficient. ROS 2's DDS communication and semantic-map sharing will allow multiple robots to collaboratively build and share a global map. Edge inference and distributed filtering algorithms will synchronize pose estimates, increasing operational efficiency and fault tolerance. Deep reinforcement learning can automatically tune filter parameters and resampling strategies, while meta-learning approaches enable quick adaptation to new environments. Over time, robots will accumulate experience to self-optimize their localization performance, achieving "zero" or "minimal" manual parameterization. As the ROS ecosystem matures and standards like OPC UA

and ROS-Industrial advance, localization modules will become more robust and reusable. Seamless, cross-vendor integration will enable deployment of high-reliability, safety-certified localization solutions across diverse verticals—pharmaceutical production, food processing, hazardous-environment operations—propelling intelligent manufacturing to the next level.

7. Conclusion

This paper presented a systematic comparison of two mainstream ROS localization algorithms—Extended Kalman Filter (EKF) and Adaptive Monte Carlo Localization (AMCL)—across open, obstructed, and dynamically disturbed industrial scenarios. Experimental results demonstrate EKF's advantage in rapid convergence and dynamic recovery, while AMCL achieves superior static accuracy in structured environments. To address their respective limitations, we proposed multi-stage hybrid strategies, visual and semantic augmentations, and online adaptive tuning. Our analysis and recommendations offer practical guidance for implementing high-precision, robust localization systems in real-world industrial robotics applications.

References

[1] Alajami, Abdussalam A., et al. "A ROS-based distributed multi-robot localization and orientation strategy for heterogeneous robots." Intelligent service robotics 16.2 (2023): 177-193.

[2] Irwansyah, Achmad Syahrul, et al. "ROS-based multi-sensor integrated localization system for cost-effective and accurate indoor navigation system." International Journal of Intelligent Robotics and Applications (2024): 1-19.

[3] Çelik, Orkan Murat, and Murat Köseoğlu. "A modified dijkstra algorithm for ros based autonomous mobile robots." Journal of Advanced Research in Natural and Applied Sciences 9.1 (2023): 205-217.

[4] Pandey, Anish, et al. "Implementation of simultaneous localization and mapping for TurtleBot under the ROS design framework." International Journal on Interactive Design and Manufacturing (IJIDeM) 18.6 (2024): 3799-3812.

[5] Sandanika, Wanni Arachchige Heshani, et al. "Ros-based multi-robot system for efficient indoor exploration using a combined path planning technique." Journal of Robotics and Control (JRC) 5.5 (2024): 1241-1260.

[6] Ahmed Abdulsaheb, Jaafar, and Dheyaa Jasim Kadhim. "Real-Time SLAM Mobile Robot and Navigation Based on Cloud-Based Implementation." Journal of Robotics 2023.1 (2023): 9967236.

[7] Gürevin, Bilal, et al. "A Novel Control and Monitoring Interface Design for ROS Based Mobile Robots." Düzce Üniversitesi Bilim ve Teknoloji Dergisi 12.1 (2024): 496-509.

[8] Bonci, Andrea, et al. "Robot operating system 2 (ros2)-based frameworks for increasing robot autonomy: A survey." applied sciences 13.23 (2023): 12796.

[9] Mărieș, Marco, and Mihai Olimpiu Tătar. "Design and Simulation of Mobile Robots Operating Within Networked Architectures Tailored for Emergency Situations." Applied Sciences 15.11 (2025): 6287.

[10] Kobayashi, Masato, and Naoki Motoi. "Bsl: Navigation method considering blind spots based on ros navigation stack and blind spots layer for mobile robot." IEEE Transactions on Industry Applications 60.1 (2023): 1695-1704.