IMPROVEMENT OF NAVIGATION AND CONTROL ALGORITHMS OF AUTONOMOUS MOBILE ROBOTS BASED ON SLAM.

assistent Norqo‘ziyev Quvonchbek Komiljon o‘g‘li
quvonchbek9535@gmail.com
assistent Ulashev Asror Nasriddinovich
ulashevasror8@gmail.com
assistenti: Aynakulov Toxir Turg‘un o‘g‘li.
aynaqulovtohir@gmail.com
Mirzo Ulug‘bek nomidagi O‘zbekiston Milliy universiteti Jizzax filiali
Dandu Jayabharath Reddy, Assistent professor , Sambhram university, Jizzax

Abstract: This paper presents an improved approach to navigation and control algorithms for autonomous mobile robots (AMRs) using Simultaneous Localization and Mapping (SLAM). SLAM enables robots to build accurate maps of their environments while simultaneously localizing themselves, making it critical for efficient and autonomous navigation in dynamic, unstructured environments. The proposed improvements focus on optimizing sensor fusion techniques, integrating data from LiDAR, cameras, and IMUs to enhance localization accuracy and map reliability. Advanced path planning algorithms, such as A* and Dynamic Window Approach (DWA), are combined with robust control methods, including Model Predictive Control (MPC) and PID controllers, to ensure precise trajectory execution and real-time obstacle avoidance. Experimental results demonstrate the effectiveness of these enhancements, leading to more reliable, efficient, and scalable AMR navigation in complex environments..

Keywords: Autonomous Mobile Robots (AMRs), Simultaneous Localization and Mapping (SLAM), Sensor Fusion, Path Planning, LiDAR, Camera, Inertial Measurement Unit (IMU), A* Algorithm, Dynamic Window Approach (DWA), Model Predictive Control (MPC), PID Control, Obstacle Avoidance, Navigation Optimization..

Introduction. Autonomous mobile robots (AMRs) are becoming an integral part of various industries due to their ability to operate independently in dynamic and unstructured environments. One of the key challenges in autonomous navigation is the robot's ability to accurately localize itself and map its surroundings, which are essential for effective path planning and obstacle avoidance. Simultaneous Localization and Mapping (SLAM) has emerged as a powerful solution for addressing this challenge. SLAM enables robots to build a map of an unknown environment while simultaneously determining their position within it.

However, despite the success of SLAM in improving navigation, there remain opportunities to enhance its performance, particularly in terms of accuracy, efficiency, and adaptability in dynamic settings. Integrating data from multiple sensors, such as LiDAR, cameras, and Inertial Measurement Units (IMUs), can improve the robot's perception of its environment, leading to more reliable localization and mapping. Additionally, optimizing path planning and control algorithms ensures that the robot can navigate complex environments with greater precision and stability.

This paper focuses on improving navigation and control algorithms for AMRs by enhancing SLAM performance through advanced sensor fusion, robust path planning methods, and efficient control strategies. By addressing the limitations of current SLAM-based approaches, the proposed improvements aim to enable more accurate, reliable, and adaptable navigation for autonomous robots in real-world applications.

Brief analysis of existing algorithms

SLAM Algorithms	Path planning Algorithms	Control Algorithms	Sensor Fusion
• **EKF-SLAM** • **FastSLAM** • **Graph-Based SLAM** • **Visual SLAM (vSLAM)**	• **A*** • **RRT (Rapidly-exploring Random Tree)** • **DWA (Dynamic Window Approach)**	• **PID Control** • **Model Predictive Control (MPC)**	• **LiDAR + IMU Fusion** • **Visual-Inertial Odometry**

SLAM algorithms

- **EKF-SLAM** - Extended Kalman Filter-based SLAM is effective for small environments with low uncertainty. However, its quadratic complexity in terms of landmarks limits scalability.
- **FastSLAM** - A particle-filter-based approach that scales better than EKF-SLAM and allows handling larger environments. It’s faster but sensitive to noise in sensor data, which can affect localization accuracy.
- **Graph-Based SLAM** - Efficient for large-scale environments by representing poses and landmarks as nodes in a graph. It achieves high accuracy but demands significant computational resources due to graph optimization algorithms.
- **Visual SLAM (vSLAM)** - Relies on camera input and image features for mapping and localization. It is affordable but performance decreases in low-light or feature-sparse environments.

Path planning algorithms

- **A*** - A heuristic search algorithm that finds the shortest path. While precise, its computational cost is high in large environments, limiting real-time performance.

- **RRT (Rapidly-exploring Random Tree)** - Efficient for high-dimensional pathfinding, especially in complex environments, but the paths generated are often suboptimal.
- **DWA (Dynamic Window Approach)** - Real-time obstacle avoidance and path planning, particularly for dynamic environments. While effective for local navigation, it may not find globally optimal paths.

Control algorithms

- **PID Control** - Simple and widely used for trajectory following but less effective in non-linear or dynamic environments.
- **Model Predictive Control (MPC)** - Optimizes control inputs based on future predictions, leading to smooth and accurate movement. However, MPC is computationally expensive, making it challenging for real-time applications.

Sensor fusion

- **LiDAR - IMU Fusion** - High-accuracy localization, particularly in structured environments, but sensitive to sensor noise and reflective surfaces.
- **Visual-Inertial Odometry** - Combines camera and IMU data to improve localization in environments where LiDAR is unavailable or less effective.

Performance charts

Below is an overview of performance factors for each algorithm. These factors include **computational efficiency**, **accuracy**, **scalability**, and **suitability for dynamic environments**.

Algorithm	Computational Efficiency	Accuracy	Scalability	Dynamic Environment Suitability
EKF-SLAM	Low	High	Poor	Low
FastSLAM	Medium	Medium	Medium	Low
Graph-based SLAM	Low	High	High	Medium
Visual SLAM	High	Medium	Medium	Low
A* Path Planning	Low	High	Poor	Low
RRT Path Planning	High	Medium	High	Medium
DWA Path Planning	High	Medium	Medium	High
PID Control	High	Medium	High	Low
MPC Control	Low	High	Medium	Medium

Mathematical Models

EKF-SLAM Model:

$$\mathbf{x}_t = f(\mathbf{x}_{t-1}, \mathbf{u}_t) + \mathbf{w}_t$$

$$\mathbf{z}_t = h(\mathbf{x}_t) + \mathbf{v}_t$$

- State: $\mathbf{x}_t$ represents the state (robot's position, orientation, etc.).
- Control: $\mathbf{u}_t$ is the control input.
- Measurement: $\mathbf{z}_t$ is the observation from sensors.
- Process and measurement noise: $\mathbf{w}_t$ and $\mathbf{v}_t$ are noise terms.

FastSLAM Model:

- FastSLAM uses particle filters:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}, \mathbf{u}_{1:t}) = \sum_{i=1}^{N} w_t^i \delta(\mathbf{x}_t - \mathbf{x}_t^i)$$

- Each particle represents a hypothesis for the robot's trajectory.

A Path Planning Model:

- Cost function:

$$f(n) = g(n) + h(n)$$

- $g(n)$: Cost from start node to the current node.
- $h(n)$: Heuristic estimate of cost from the current node to the goal node.

RRT Path Planning Model:

- Rapidly explores the configuration space by randomly generating sample points and expanding a tree:

$$\mathbf{T} \leftarrow \mathbf{T} \cup \{\text{random sample, nearest node}\}$$

PID Control Model:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt}$$

- $e(t)$ is the error between the desired and current state.
- K_p, K_i, K_d are the proportional, integral, and derivative gains, respectively.

Model Predictive Control (MPC):

- Optimization problem:

$$\min_{u_t} \sum_{k=0}^{N-1} \left[x_k^\top Q x_k + u_k^\top R u_k\right]$$

- Subject to constraints on control inputs u_k and system dynamics. Q and R are weight matrices.

Our proposed FusionSLAM-RRT algorithm

FusionSLAM-RRT is an algorithm designed to help **autonomous robots** (robots that move on their own) figure out where they are, build a map of their surroundings, and safely navigate through spaces without bumping into things—even if those things are moving!.

How does it work?

1. **Seeing and Sensing the Environment (Sensor Fusion)**:
   - The robot uses **three kinds of sensors** to understand its surroundings:

- **LiDAR** (like a laser scanner) to measure distances to walls or objects.
- **Cameras** to see what's in front of it.
- **IMU** (like an internal compass) to know how it's moving.

o By combining these sensors, the robot can "see" better and more accurately, even if one of the sensors has trouble (like in a dark room where the camera can't see well).

2. **Building a Map and Knowing Where It Is (SLAM)**:

o As the robot moves, it **builds a map** of its surroundings and tracks where it is on that map (like drawing a map while walking through a maze and marking your location).

o The robot also **tracks moving objects** (like people walking), so it knows if something is coming into its path.

3. **Planning the Path (RRT Path Planning)**:

o The robot uses an algorithm called **RRT** to figure out the best path to its destination. This is like drawing a path through a maze to get to the exit.

o If something moves in front of the robot, it can quickly **re-plan its path** in real-time to avoid obstacles.

4. **Moving Smoothly (Control)**:

o The robot has two ways of controlling its movement:

- **PID Control** for small, careful adjustments (like when you're walking slowly and need to make tiny turns).
- **MPC (Model Predictive Control)** for bigger adjustments and long-range planning (like when you see an obstacle far away and plan a smooth turn to avoid it).

5. **Avoiding Obstacles (Smart and Safe Movement)**:

o The robot doesn't just react to obstacles at the last moment. It **predicts** where moving objects (like people) will go and avoids them before they get too close. This makes the robot's movement smarter and safer.

Mathematical models FusionSLAM-RRT algorithm

The robot combines data from multiple sensors (LiDAR, camera, IMU) to understand its position and surroundings more accurately. To do this, we use the **Extended Kalman Filter (EKF)** to process the noisy sensor data and produce a better estimate of the robot's position and map.

EKF Prediction Step:

$$\hat{x}_t = A\hat{x}_{t-1} + Bu_t + w_t$$

Where:

- $\hat{x}_t$ is the robot's estimated position at time t,
- A is the system transition model (how the robot moves from one state to another),
- B is the control input model,
- u_t is the control input (commands given to the robot, like speed and direction),
- w_t is the process noise (uncertainty from sensors).

EKF Update Step:

$$\hat{z}_t = C\hat{x}_t + v_t$$

Where:

- $\hat{z}_t$ is the sensor measurement (data from LiDAR, camera, IMU),
- C is the observation model (how the sensor reads the environment),
- v_t is the measurement noise (sensor errors).

This filtering process helps the robot combine and interpret all the sensor data to get a clearer understanding of its position and environment.

SLAM (Simultaneous Localization and Mapping)

As the robot moves, it builds a map of its surroundings while figuring out where it is on that map. In SLAM, the robot continuously updates both its position and the map using a mathematical representation.

State Update: The robot's state (position and orientation) is updated as it moves:

$$x_{t+1} = f(x_t, u_t, w_t)$$

Where:

- x_{t+1} is the new state (robot's position at time $t+1$),
- $f(x_t, u_t)$ is the motion model that predicts the robot's next state based on its current state and control input,
- w_t is noise or uncertainty.

Map Update:

As the robot moves and gathers new information, it updates its internal map M_t:

$$M_{t+1} = M_t \oplus \Delta M_t$$

Where:

- M_t is the current map,
- ΔM_t is the new information gathered from sensors.

The map is updated incrementally as the robot moves and gathers more data.

Path Planning (RRT with Dynamic Biasing)

For finding a path, the robot uses the **Rapidly-exploring Random Tree (RRT)** algorithm. RRT generates a tree of possible paths by randomly exploring the space, trying to connect the start point to the goal.

Tree Expansion:

At each step, the algorithm picks a random point q_{rand} in the environment and connects it to the nearest point $q_{nearest}$ in the tree:

$$T_{new} = T_{old} \cup \{q_{rand}, q_{nearest}\}$$

Where:

- T_{old} is the current tree,
- q_{rand} is the randomly chosen point,
- $q_{nearest}$ is the nearest existing point in the tree.

The algorithm expands the tree towards q_{rand} until it eventually reaches the goal.

Dynamic Replanning:

If the robot detects a moving obstacle in its path, it dynamically replans the path by updating the RRT tree in real-time. It biases the tree towards unexplored areas to avoid collisions:

$$T_{new} = T_{old} \cup \{\text{new safe point}, q_{nearest}\}$$

The robot continues updating the tree as it moves, ensuring it avoids obstacles and reaches its goal safely.

Control System (PID and MPC)

PID Control (for small adjustments):

The **PID controller** keeps the robot on track by minimizing the error between the desired path and the robot's current position.

The control output $u(t)$ is calculated as:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt}$$

Where:

- $e(t)$ is the error (difference between the desired and actual position),
- K_p, K_i, K_d are the proportional, integral, and derivative gains.

MPC (Model Predictive Control) (for bigger movements):

MPC predicts the robot's future states based on a model of its motion and optimizes the control inputs accordingly.

The MPC solves the following optimization problem:

$$\min_{u_t} \sum_{k=0}^{N-1} \left[x_k^\top Q x_k + u_k^\top R u_k \right]$$

Where:

- x_k are the predicted future states,
- u_k are the control inputs to minimize the cost,
- Q and R are weight matrices that determine the importance of staying on track and minimizing control effort.

Obstacle Avoidance (Predicting Future Positions)

To avoid collisions, the robot predicts the future positions of moving obstacles using a velocity-based model:

$$p_{future} = p_{current} + v \cdot \Delta t$$

Where:

- p_{future} is the predicted future position of the obstacle,
- $p_{current}$ is the current position of the obstacle,
- v is the velocity of the obstacle,
- Δt is the time step for prediction.

The robot adjusts its path accordingly to avoid these predicted positions.

Conclusion of the mathematical model

- **FusionSLAM-RRT** uses **sensor fusion** to combine different sensors for better localization and mapping.
- The **SLAM model** updates both the robot's position and the map of the environment.
- **RRT** explores the environment dynamically and replans the path in real-time when new obstacles are detected.
- **PID and MPC** control systems ensure smooth movement along the path.
- **Predictive obstacle avoidance** ensures the robot can safely navigate even in dynamic environments.

Conclusion

The **FusionSLAM-RRT** algorithm effectively combines sensor fusion, SLAM, and dynamic path planning to enable autonomous mobile robots to navigate complex and changing environments. By integrating LiDAR, camera, and IMU data, it ensures accurate localization and mapping. The RRT-based path planning dynamically adjusts to obstacles, while the PID and MPC control systems maintain smooth, precise movement. Predictive obstacle avoidance further enhances safety. This approach improves real-time navigation performance, making it well-suited for autonomous robots operating in both structured and dynamic environments.

References

1. Toxir Turg‘un o‘g, A., & Komiljon o‘g, N. Z. Q. (2024). CALCULATION OF SPECIAL FUNCTIONS WITH ARBITRARY PRECISION IN THE. NET FRAMEWORK ENVIRONMENT. *Лучшие интеллектуальные исследования*, *19*(3), 142-149.

2. Komiljon o‘g, N. Z. Q., Hasan o‘g‘li, T. A., & Sodiq o‘g‘li, S. D. (2024). AI WAYFINDING ALGORITHM OF A MOBILE ROBOT. *Лучшие интеллектуальные исследования*, *19*(3), 150-156.

3. Komiljon o‘g, N. Z. Q., & Tohir Turg'un o'gli, A. (2024). A FAST AND OPTIMAL WAYFINDING ALGORITHM USING INFORMATION FROM LIDAR SENSORS IN THE ENVIRONMENT. *Лучшие интеллектуальные исследования*, *19*(3), 157-163.

4. Norqo‘ziyev, Q., Aynakulov, T., Mahkamov, S., & Sodiqov, D. (2024). IMPLEMENTING MOBILE ROBOT WALKING PATH CONTROL AND TRACKING IN C# WINDOWS FORM APPLICATION. *International Journal of scientific and Applied Research*, *1*(3), 95-99.

5. Норкозиев, К., & Тоджиев, А. (2023). Использование искусственных нейронных сетей при разработке алгоритма поиска оптимального пути мобильных роботов в динамических средах. *Информатика и инженерные технологии*, *1*(1), 25-29.

6. Norqo'ziyev, Q. (2023). MOBIL ROBOTLAR UCHUN YO'LNI REJALASHTIRISH ALGORITMI. *Research and implementation*.

7. Mirzaev, N. M., Meliev, F. F., & Baratov, J. R. (2022). ARTIFICIAL INTELLIGENCE TECHNOLOGIES TO DETECT CANCERS FROM HUMAN COLON HISTOLOGICAL IMAGES. ББК 22.18 С 56, 181.

8. Жомуродов, Д. (2024). Оптимизация и улучшение производительности алгоритма шифрования Blowfish. Новый Узбекистан: наука, образование и инновации, 1(1), 155-158.

9. Жомуродов Д. М., Улашев А. Н., Мелиева М. З. ЭФФЕКТИВНОЕ ОБУЧЕНИЕ МАССИВАМ: ПРИМЕНЕНИЕ VR/AR И МАТЕМАТИЧЕСКИХ МОДЕЛЕЙ //SAMARALI TA’LIM VA BARQAROR INNOVATSIYALAR JURNALI. – 2024. – Т. 2. – №. 5. – С. 1011-1023

10. Tojiyev, A., A. Ulashev, and Botir Abdualimov. "WINDOWS FORM ILOVASIDA MATEMATIK FUNKSIYALAR GRAFIGINI YASASH." International Scientific and Practical Conference on Algorithms and Current Problems of Programming. 2023.

11. Abdullayevich U. X. ESSENCES OF ALGORITHMIC COMPETENCE OF FUTURE ICT SPECIALISTS IN HIGHER EDUCATION //Thematics Journal of Education. – 2022. – Т. 7. – №. 4.

12. Akhatov , A., & Ulugmurodov , S. A. (2023). BRAILLE METHODS OF TEXT RECOGNITION USING "DEEP LEARNING". SCIENCE AND INNOVATIVE DEVELOPMENT, 6(1), 32–41. Retrieved from https://ilm.mininnovation.uz/index.php/journal/article/view/387

13. Akmal Akhatov, & Shokh Abbos Ulugmurodov. (2023). TRAINING DATA SELECTION AND LABELING FOR MACHINE LEARNING BRAILLE RECOGNITION MODELS. International Journal of Contemporary Scientific and Technical Research, (Special Issue), 15–21. Retrieved from https://journal.jbnuu.uz/index.php/ijcstr/article/view/326

14. Turakulov Olim Xolbo‘tayevich, Kayumov Oybek Achilovich, & Kayumov Obid Achilovich. (2022). INTERAKTIV INTELLEKTUAL ELEKTRON TA’LIM RESURLARINI QO‘LLASH ORQLI AXBOROT TEXNOLOGIYALARI SOHASIGA KADRLAR TAYYORLASHNING SIFATINI OSHIRISH. International Journal of Contemporary Scientific and Technical Research, 1(1), 348–351. Retrieved from https://journal.jbnuu.uz/index.php/ijcstr/article/view/87

15. Kayumov , O. A. (2022). INTERAKTIV INTELLEKTUAL ELEKTRON TA’LIM RESURSIDAN O‘QUV VA MUSTAQIL ISH JARAYONILARIDA FOYDALANISH METODIKASI. RESEARCH AND EDUCATION, 1(2), 14–21. Retrieved from https://researchedu.org/index.php/re/article/view/596