Autonomous Robot Systems for Item Retrieval

Overview

This project focuses on developing an autonomous robotic system to efficiently retrieve and sort items in a simulated environment using ROS2. The solution integrates item detection, autonomous navigation, and multi-robot coordination to maximize task efficiency while adhering to constraints such as obstacle avoidance and zone-specific item placement.

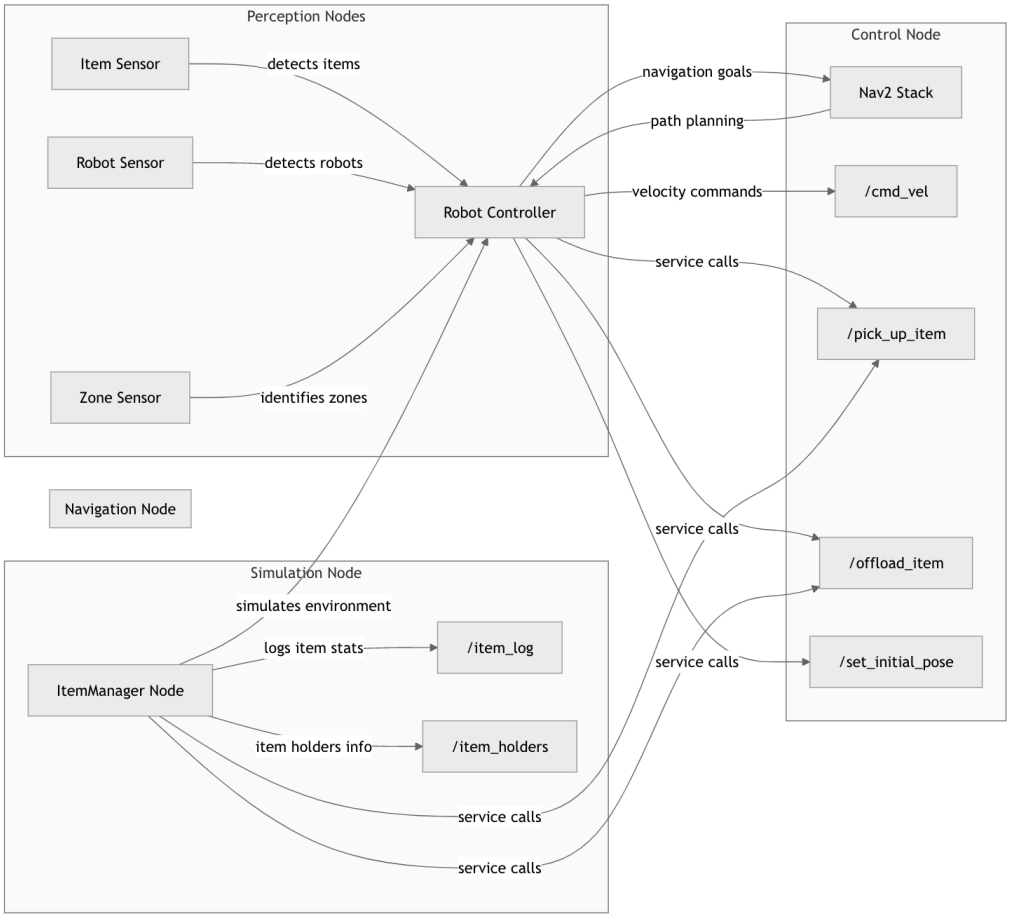
The primary methodology involved modular design and leveraging ROS2's Nav2 stack for navigation. The system employs a combination of perception (camera-based item detection) and control strategies (state machines) to ensure smooth operations. For scalability and robustness, the implementation supports up to three TurtleBot3 Waffle Pi robots, enabling collaborative item retrieval.

The choice of methodologies was influenced by the need for reliability and real-time decision-making in a dynamic environment. Key design decisions, such as utilizing a subscription-publisher model for data flow and employing pre-defined zones for item sorting, were made to ensure deterministic behavior and minimize conflicts in multi-robot setups.

By systematically integrating these functionalities, the solution provides a comprehensive framework for item retrieval that can be extended for real-world applications such as warehouse automation or disaster recovery scenarios.

Architecture

The architecture of the autonomous robotic system is designed to efficiently detect, retrieve, and deposit items in a simulated environment using ROS2. The architecture, as illustrated in the diagram, integrates various nodes to ensure seamless perception, control, navigation, and simulation.



Key Components

1. **Perception Nodes:**
    - **Item Sensor:**Responsible for detecting items and publishing their attributes (e.g., position, size, color) to the **/robot_name/items** topic.
    - **Robot Sensor:** Identifies other robots in the environment and publishes their positions to the **/robot_name/robots** topic.
    - **Zone Sensor**: Detects zones in the environment and publishes their attributes to the **/robot_name/zone** topic.

    These nodes provide essential environmental data, which is processed by the Robot Controller to make decisions.

2. **Control Node:**
    - **Robot Controller:** Central to the system, it subscribes to topics published by perception nodes and uses this data to navigate, pick up, and offload items. It interfaces with:
        - **Nav2 Stack**: For path planning and goal navigation.
        - **/cmd_vel**: To issue velocity commands for robot movement.
        - **Services:**
            - **/pick_up_item:** To pick up items.
            - **/offload_item:** To offload items at designated zones.
            - **/set_initial_pose:** To initialize the robot's position.

3. **Navigation Node:**
    - **Nav2 Stack:** Facilitates autonomous navigation by providing path planning capabilities and interacting with the Robot Controller.

4. **Simulation Node:**
    - **ItemManager Node:** Simulates the environment by managing item locations and zones. It publishes:
        - **/item_log:** Logs the status of collected items.
        - **/item_holders:** Tracks which items are held by which robot.
    - Provides services like **/pick_up_item** and **/offload_item**.

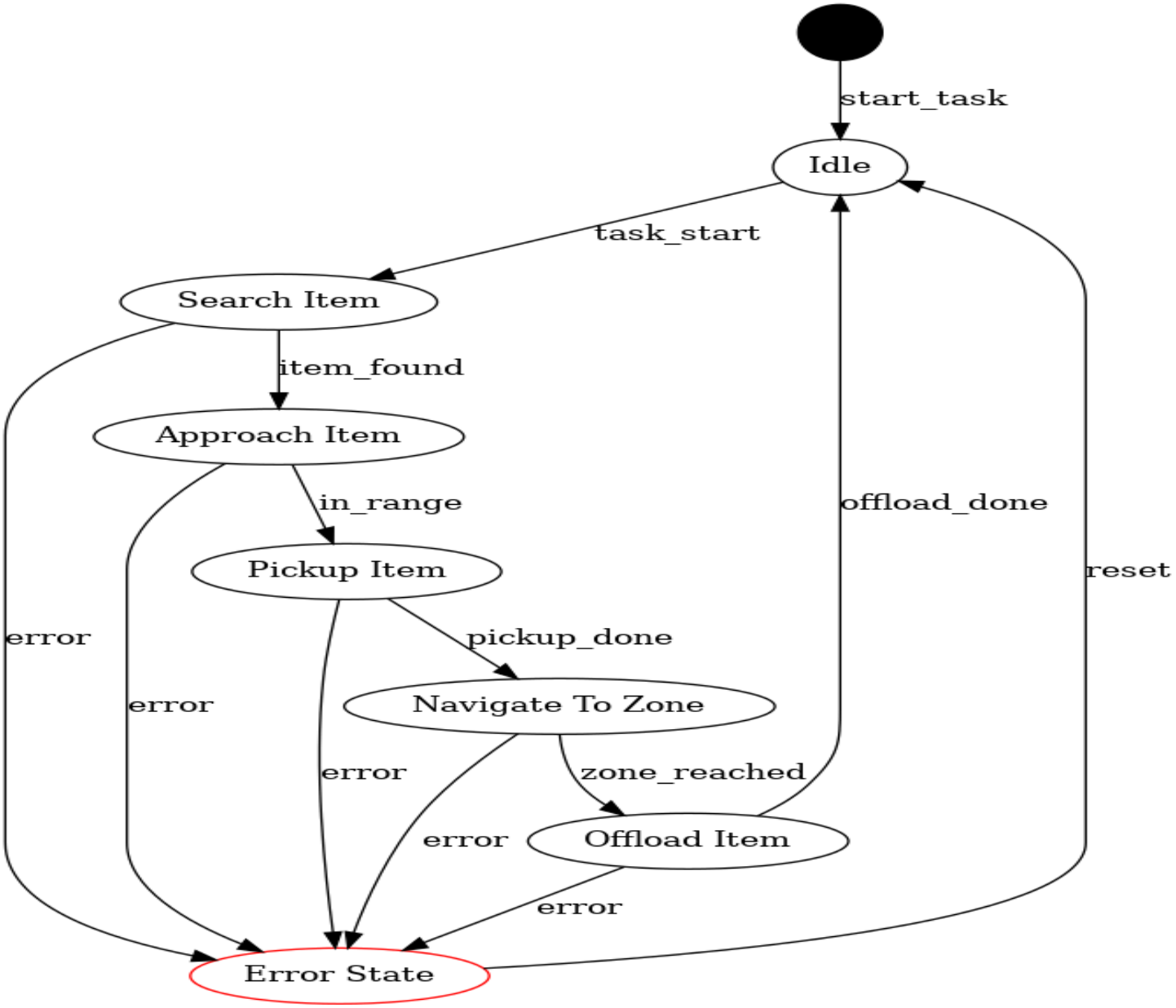
Integration

The Robot Controller acts as the system's hub, processing inputs from perception nodes and commanding outputs to control and simulation nodes. This modular architecture ensures scalability, allowing multiple robots to operate collaboratively without conflict.

This diagram effectively demonstrates the interconnected nature of nodes, highlighting their responsibilities and communication flows to achieve robust task performance in a dynamic environment.

Control

The state machine provided illustrates the control logic of the **Robot Controller Node,** which is central to managing the autonomous operation of the robotic system. This control architecture ensures that tasks such as item detection, pickup, navigation, and offloading are executed in a structured and reliable manner, while also incorporating robust error handling.



State Machine Description

1. **Idle:** The system begins in the **Idle** state, awaiting task initialization. This state serves as the default starting point and ensures that the robot is ready to proceed when a task is assigned.
2. **Search Item:** Upon task initiation, the robot transitions to the **Search Item** state, where it uses sensory data to locate items within the environment. The transition to the next state occurs when an item is detected (**item_found**).
3. **Approach Item:** After detecting an item, the robot moves to the **Approach Item** state, where it calculates and adjusts its trajectory to move within range of the item. The transition to the next state is triggered by reaching the appropriate proximity (**in_range**).
4. **Pickup Item:** In this state, the robot interacts with the **/pick_up_item** service to collect the detected item. Upon successful pickup, the robot transitions to the **Navigate To Zone** state (**pickup_done**).
5. **Offload Item:** At the designated zone, the robot interacts with the **/offload_item** service to deposit the collected item. Upon successful offloading, the robot transitions back to the **Idle** state (**offload_done**).

Error Handling

The state machine incorporates a dedicated **Error State** to handle failures (e.g., navigation errors, service failures). Any state can transition to **Error State** (**error**) in the event of a failure. From **Error State**, the robot can reset and return to **Idle** (**reset**), ensuring continued functionality.

Appropriateness and Design

This control architecture is well-suited for the task, as it ensures a systematic approach to task execution while providing error recovery. The modular design facilitates clarity and scalability, making it adaptable to dynamic environments and supporting multiple robots. The state machine effectively balances task execution with robust error handling, aligning with the project objectives.

Evaluation

1. Simulation-Based Evaluation

The system was evaluated in a Gazebo simulation under three configurations: single robot, two robots, and three robots. Each scenario ran for 120 seconds, focusing on task completion time, throughput, success rates, and collision avoidance.

Table 1 summarizes the results:

Table 1: Simulation Results

Scenario	Avg. Task Time (sec)	Total Items Delivered	Total Value	Pick-up Rate (%)	Delivery Rate (%)	Collisions	Total Time (In Sim Time)
Single Robot	24	5	25	100	100	0	120 seconds
Two Robots	12	10	75	100	100	0	120 seconds
Three Robots	7.5	16	165	100	100	0	120 seconds

Key Observations

1. **Efficiency and Throughput**: Adding more robots significantly improved efficiency. The average task time decreased from 24 seconds (single robot) to 7.5 seconds (three robots), while throughput increased from 5 items delivered to 16 items delivered in the same time frame.
2. **Collision Avoidance**: No collisions were observed, highlighting robust obstacle avoidance in all configurations.
3. **Success Rates**: Pick-up and delivery success rates remained at 100% across all scenarios, demonstrating high reliability.

2. Descriptive Evaluation

The system was evaluated qualitatively by observing the robots' behaviors during simulation runs with one, two, and three robots. This evaluation focused on navigation smoothness, task division, and item detection accuracy.

Single-Robot Configuration

- The robot navigated directly to targets without unnecessary detours and accurately delivered items to the correct zones.
- After offloading, it efficiently identified and navigated to the next closest target, ensuring minimal idle time.

Two-Robot Configuration

- The robots collaborated effectively by prioritizing specific item colors (e.g., one handled red items, the other green or blue).
- Both robots followed smooth and precise paths, minimizing unnecessary detours and avoiding collisions.
- After completing a task, each robot independently identified its next target and returned to its designated offloading zone.

Three-Robot Configuration

- The robots divided tasks by color, synchronizing efficiently to manage items and zones without conflicts.
- The robots demonstrated synchronized task execution, ensuring smooth operations without conflicts or excessive waiting times.
- Navigation paths remained optimal, with minimal unnecessary detours. All robots successfully picked up items and delivered them to the correct zones.

Strengths Observed

1. **Navigation Smoothness:** Robots efficiently navigated to targets and zones with effective motion planning.
2. **Task Collaboration**: Multi-robot setups divided tasks by item color, reducing overlaps and boosting efficiency.
3. **Accuracy:** All detected items were correctly picked up and delivered to designated zones.
4. **Scalability:** The system maintained smooth operations as the number of robots increased.

Limitations

- Minor unnecessary detours occurred in multi-robot scenarios due to obstacle avoidance, but these had negligible impact on overall performance.

3. Experimental Evaluation

The experimental evaluation tested the system's performance and adaptability under various configurations and environmental conditions. The focus was on assessing the impact of parameter changes on the system's functionality.

Scenarios Tested

1. **Random Seeds:**
   - Different random seeds were used to vary item placement and environmental conditions.
   - **Observation:** The system consistently detected items and zones across all seeds, demonstrating robust perception and navigation.
2. **Odometry Source:**
   - Odometry was switched between encoder-based and world-based configurations.
   - **Observation:** The system maintained accurate navigation and delivery in both configurations, showcasing adaptability to different odometry sources.
3. **Zone Configurations and Deactivations (Multiple Robots):**
   - Simulations were run with 2 or 3 robots, with predefined zones adjusted for color-based delivery. In some cases, one or more zones were deactivated while multiple robots were active.
   - **Observation:** Despite the deactivation of one or more zones, active robots successfully delivered items to the remaining operational zones. This demonstrates that the system can handle certain zone deactivations while maintaining successful task execution across multiple robots.
4. **Zone Deactivation (Predefined Zone Dependency):**
   - Scenarios were tested where the predefined zone for a robot was deactivated while the robot remained active.
   - **Observation:** Robots failed to reassign tasks or find alternative zones when predefined zones were deactivated, leading to incomplete tasks and limited adaptability.

Justification:

The chosen evaluation methods—simulation-based, descriptive, and experimental—assess the system's performance, behavior, and adaptability. Simulation offers quantitative benchmarks, descriptive evaluation highlights robot collaboration, and experimental evaluation ensures robustness under varied conditions. Together, they provide a comprehensive assessment.

Safety and Ethics

Deploying autonomous robotic systems comes with important safety and ethical considerations that must be addressed for their responsible and reliable use in real-world environments.

Safety Implications

Safety is always the top priority, especially when robots operate around people and in unpredictable environments. While the system showed excellent collision avoidance in simulations, real-world scenarios are more complex. There might be unexpected obstacles, sudden human movements, or environmental changes that the robot must handle. To prevent accidents, the system should have strong sensors, backup systems, and the ability to recover from errors like power outages or communication failures. These safeguards will make the robot more reliable and safe to use.

Ethical Implications

Ethically, issues like accountability, privacy, and job displacement come into play. It's essential to define who is responsible for a robot's actions, especially if something goes wrong. Sensors used for navigation could inadvertently collect sensitive data, raising privacy concerns. To address this, the system must follow strict data protection laws, like GDPR, and be transparent about how it handles information. Additionally, the increased use of autonomous robots in industries might affect jobs, so it's vital to consider solutions like retraining programs to support affected workers.

Real-World Considerations

If implemented in the real world, the system needs rigorous testing to ensure reliability in unpredictable scenarios. Navigation and task management algorithms should work seamlessly, even in challenging conditions. Ethical concerns, such as privacy and fairness, require thoughtful system design and open communication with stakeholders. Involving the community and ensuring the technology is accessible and equitable can help the system align with broader societal values.

By carefully addressing these safety and ethical challenges, this solution can become a responsible and effective tool in real-world applications.

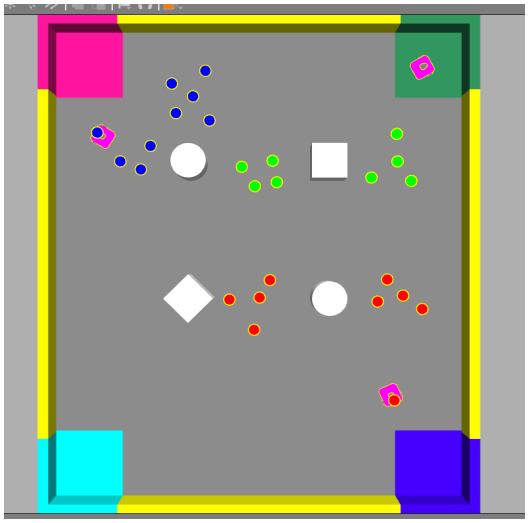
Simulation Scenarios

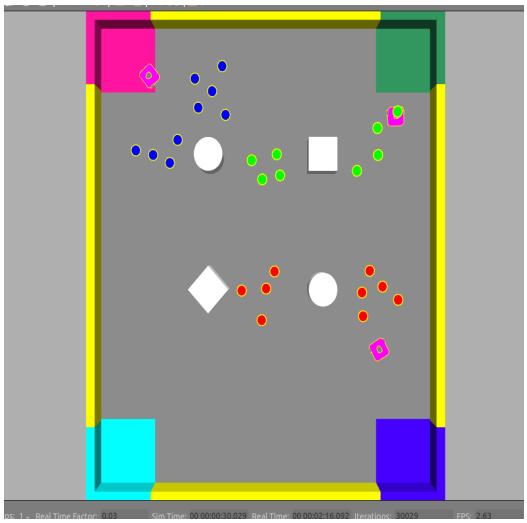
General Introduction

The simulation scenarios were designed to evaluate the system's adaptability, efficiency, and reliability under various conditions. By testing different configurations, zone availability, and environmental changes, the scenarios demonstrate how the system handles key challenges such as task coordination, navigation accuracy, and scalability.

1. **3 Robot Scenario**

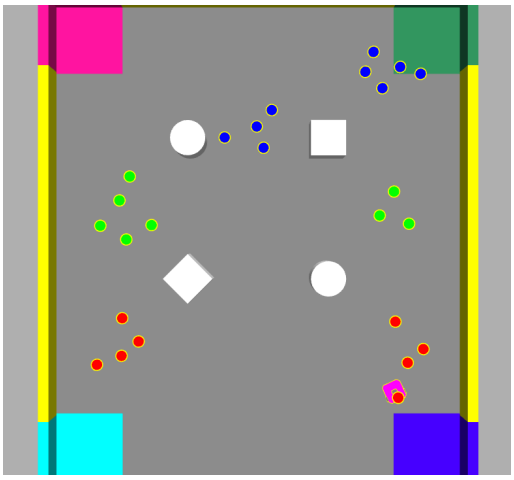
   The simulation with three robots demonstrated efficient task division by item color, smooth navigation, and collision-free collaboration. Robots avoided overlap, maximizing throughput and delivering items safely to the correct zones.





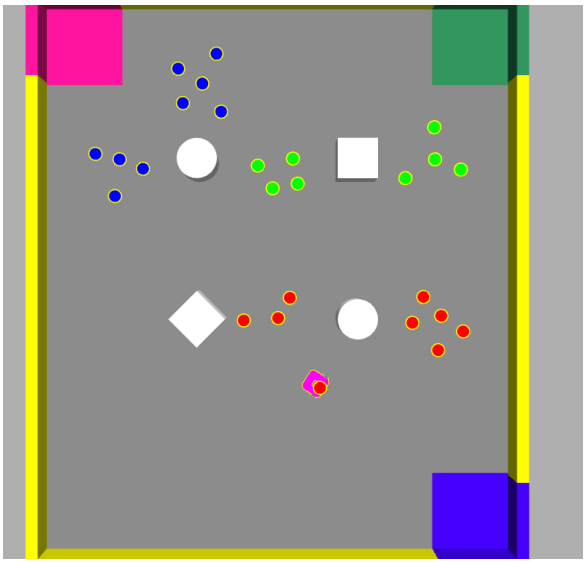
2. **Random Seed Variation Scenario**

   This scenario involved changing the random seed values to vary item placement and test adaptability. The robot efficiently detected and navigated to new item configurations, maintaining smooth operation and task completion. The system demonstrated robustness in handling dynamic changes in the environment.



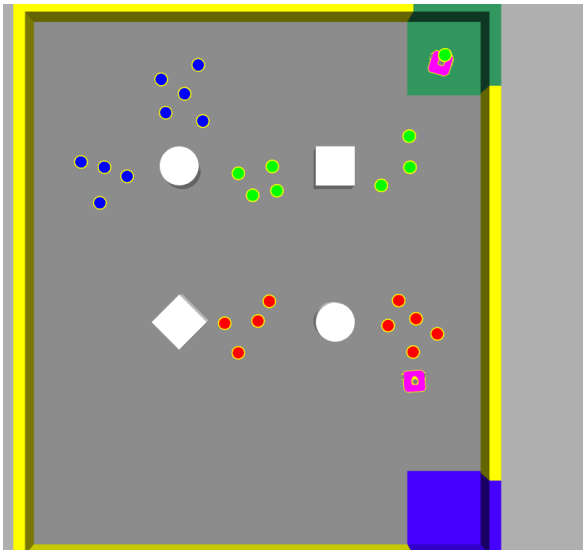
3. **Zone Deactivation Scenario (Single Robot)**

   In this scenario, the bottom-left zone was deactivated to test the system's ability to handle reduced operational zones. The single robot efficiently redirected tasks to the remaining active zones, maintaining successful item collection and delivery. This demonstrates the system's adaptability under constrained conditions.



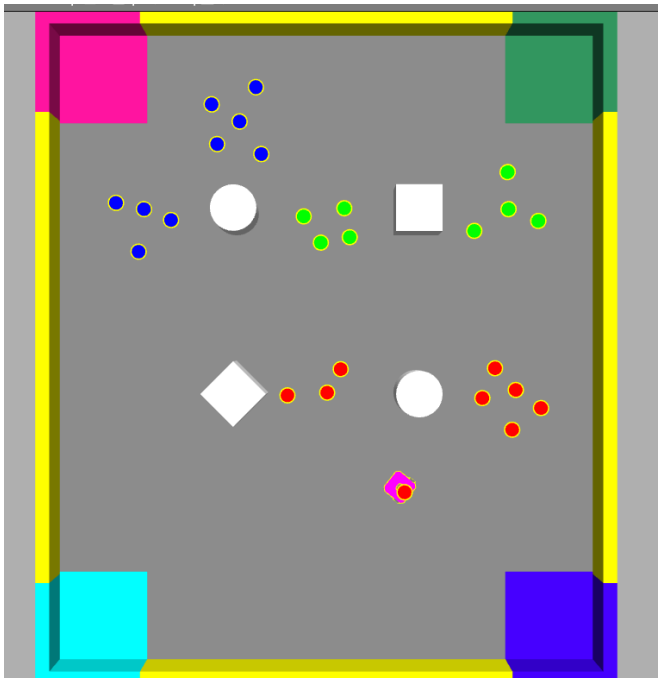
4. **Zone Deactivation Scenario (Two Robots)**

   This scenario deactivated both the bottom-left and top-left zones to test multi-robot coordination under reduced zone availability. The two robots effectively collaborated to handle tasks, redirecting items to the remaining active zones. Despite fewer delivery zones, task efficiency and collision-free operation were maintained, demonstrating adaptability in constrained environments.



5. **Odometry Source Variation Scenario**

   This scenario involved changing the odometry source from encoder-based to a world-based configuration to evaluate adaptability. The robot successfully maintained accurate navigation and task execution despite the change, demonstrating robust handling of different odometry inputs and environmental feedback systems.



Justification for the Choice of Scenarios

These scenarios were chosen to test the system's ability to handle dynamic environments, varying task complexities, and operational constraints. Each scenario focuses on a specific aspect of the system, such as multi-robot collaboration, adaptability to changes, and robustness in navigation, ensuring a comprehensive evaluation of the overall strategy.

References

[1] I. Ibrahimi, “Integrating Real-Time Object Detection with LiDAR Data for Enhanced Robotic Autonomous Navigation - Webthesis,”

Polito.it, Dec. 2024, doi: https://webthesis.biblio.polito.it/secure/33908/1/tesi.pdf.

[2] “ROS 2 Documentation — ROS 2 Documentation: Rolling documentation,” *Ros.org*, 2025. https://docs.ros.org/en/rolling/.

[3] A. K. M J, A. V. Babu, S. Damodaran, R. K. James, M. Murshid, and T. S. Warrier, “ROS2 - Powered Autonomous Navigation for TurtleBot3: Integrating Nav2 Stack in Gazebo, RViz and Real-World Environments,” *2024 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES)*, pp. 1–6, Sep. 2024, doi: https://doi.org/10.1109/spices62143.2024.10779642 .