AURO Coursework

1 Overview

The methodology employed for this task utilises a single TurtleBot3 robot with a hybrid control architecture. Control is implemented through a single composite finite state machine (Figure 2) within a single ROS 2 node robot_controller.py (Figure 1). The robot controller leverages the Nav2 Simple Commander for SLAM and navigation during the initialisation, searching, and delivery processes, forming a predominantly deliberate control architecture. In contrast, a reactive approach is applied during the approaching items process.

These design choices were influenced by the need for reliable collision avoidance and navigation, within the constraints of the available hardware and simulation environment.

2 Architecture

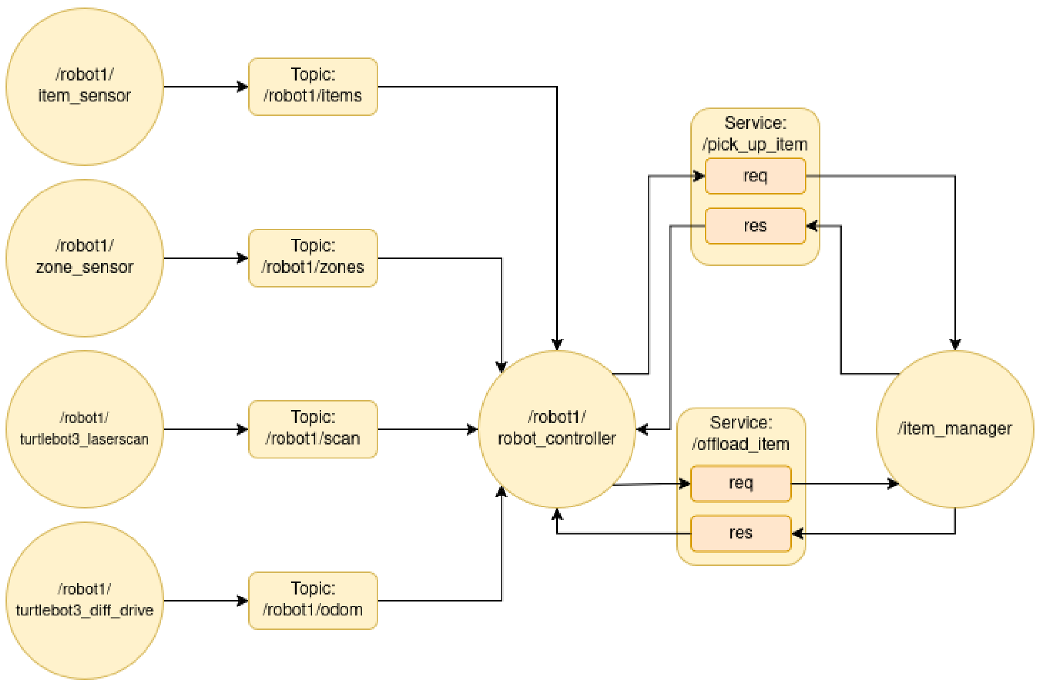

Figure 1: ROS2 High-level Architecture

As seen in Figure 1, the robot controller is the node responsible for the control strategy. The robot controller receives the latest sensor information such as items detected by the camera, zones detected by the camera, lidar data and odometry data from subscriptions to the nodes: item sensor, zone sensor, turtlebot3 laserscan and turtlebot3 diff drive, respectively via topics items, zones, scan, odom, shown in the diagram.

The robot controller also utilises the services pick up item and offload item provided by the item manager node, to pick up and offload items as needed throughout the task when retrieving and delivering items to their appropriate zone.

Within the robot controller, the Nav2 simple commander is used to implement successful navigation between locations, allowing for straightforward and reliable implementation of navigation.

Its important to note that there are many other nodes, topics, services and action servers required for the implementation of the solution, however these have been abstracted away to allow for a high level architecture diagram.

2.1 Control

The control architecture is organised into four primary states: Initialising, Searching, Approaching, and Delivering, each comprising composite sub-states managed through internal state machines. This hierarchical decomposition facilitated a structured and manageable problem-solving approach.

In the Initialising state, the robot first identifies all valid drop-off zones within the environment. It sequentially navigates to each potential zone location, verifies its presence via the zones topic, and proceeds until all locations are inspected. Subsequently, in the Assigning Zones sub-state, a unique mapping is established between each item colour and its corresponding drop-off zone. This mapping ensures efficient navigation during the delivery phase.

The Searching state employs a coverage strategy whereby the robot navigates to predefined search waypoints, performing a complete 360° scan at each location. If no items are detected, it advances to the next waypoint. However, this process can be interrupted: upon detection of an item, the system transitions immediately to the Approaching state.

In Approaching, the robot targets the detected item with the largest apparent diameter, using it as an estimate for proximity. This method, inspired by Millard et al [1], guides the robot until it reaches a threshold distance of 0.35 m, at which point it halts and issues a pick-up command. The Delivering state leverages the initial colour–zone mapping to navigate to the correct drop-off location. Upon arrival, the item is deposited, and the system returns to the Searching state, thereby iterating the retrieval–delivery cycle until task completion.

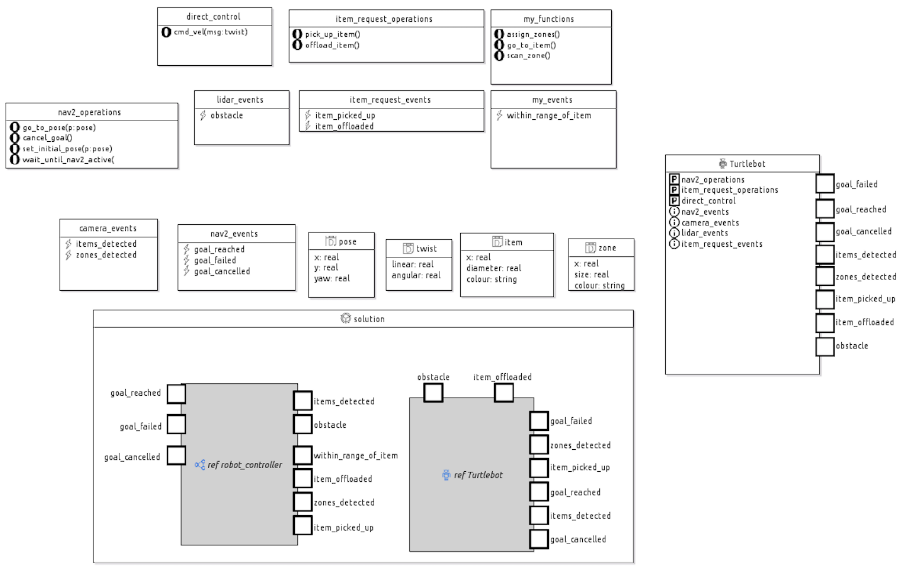

Figure 2: Robochart Finite State Machine: part 1

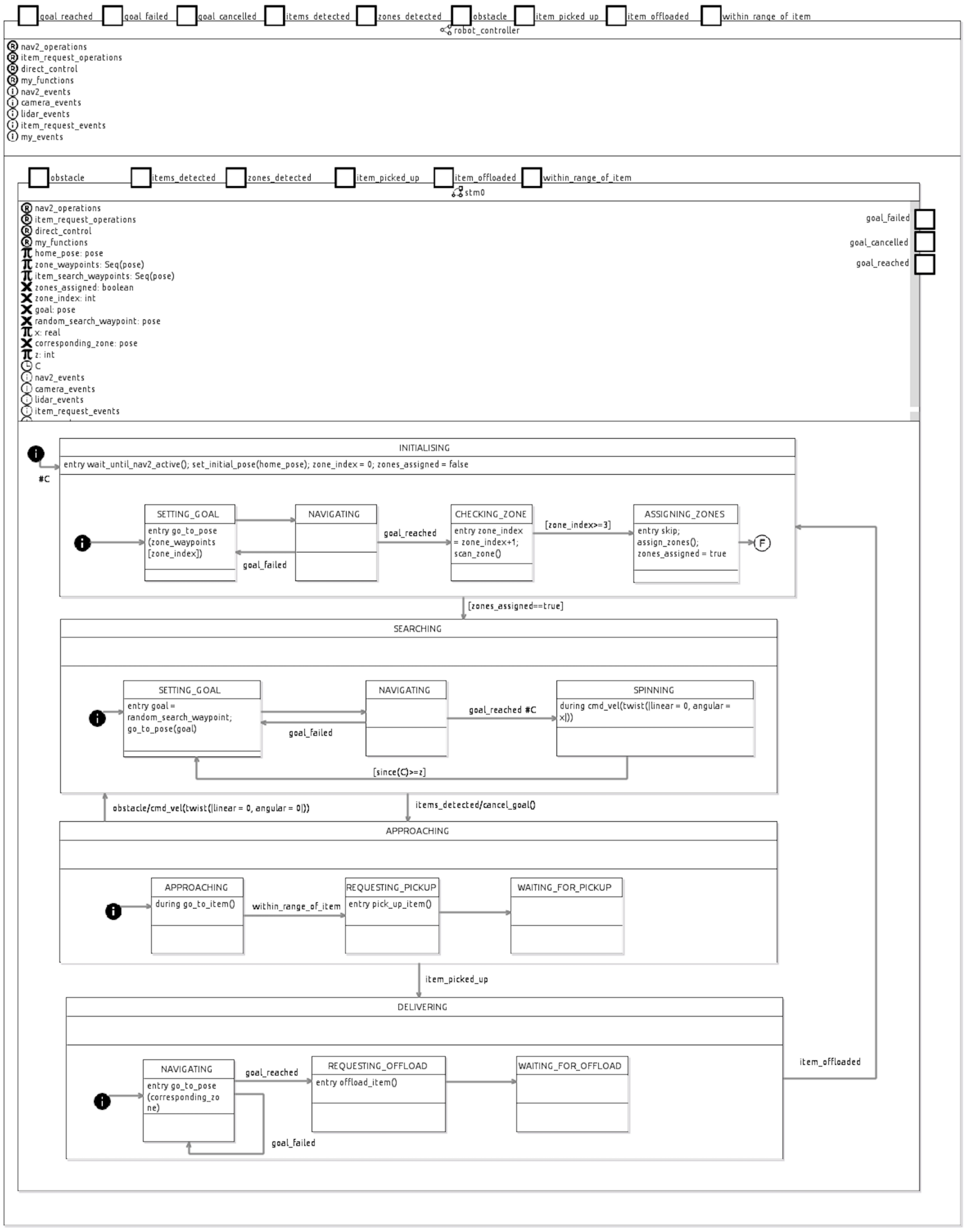

Figure 3: Robochart Finite State Machine: part 2

3 Evaluation

In the following, the solution will be evaluated using descriptive, simulation and complexity analysis. This combination ensures that the approach, performance and scalability of the solution can be evaluated respectively.

3.1 Descriptive Analysis

The solution employs a pre-execution zone-to-colour assignment, enabling direct Nav2 navigation to drop-off locations during delivery and avoiding time wasted searching for the correct zone. Although this introduces an initial delay compared to a purely reactive zone search approach, the benefits become more pronounced over longer task durations. The current method assigns zones randomly and assumes they are in the environment's corners. In cases where items are clustered, efficiency could be improved by calculating the average Euclidean distance between each colour group and each zone, then assigning zones to minimise travel distance. Such optimisation would require accurate item pose estimation and tracking, which are not supported by the current hardware. A further limitation is the absence of an item tracking mechanism to store detected items with their colours and positions. This prevents the use of route optimisation and constrains the system to a reactive strategy, where the robot navigates to the largest visible item. Initial trials of Newans' [2] pose estimation method achieved acceptable accuracy only within 1 m and required the robot to stop, centre the item in view, and synchronise with low-frequency amcl_pose updates. Given these constraints, the simpler reactive approach was retained for reliability. Hardware upgrades, such as depth-capable cameras, could make robust tracking feasible in future iterations. The single-robot configuration prioritises simplicity and reliability over throughput. Multi-robot systems could increase productivity but would require complex coordination, for example through a traffic manager node subscribing to each robot's amcl_pose, assigning navigation priorities, and issuing stop/resume commands to prevent conflicts. While such a system could enhance efficiency, it would also add communication overhead, synchronisation demands, and potential failure points. The single-robot design avoids these risks, lowers hardware costs, and ensures predictable behaviour in constrained environments.

3.2 Simulation

In simulation, the solution performed effectively due to the combination of a reactive collection strategy and Nav2 for navigation between search locations and delivery zones. This integration provided reliable obstacle avoidance and consistent path planning, maintaining task success across varied environment layouts and obstacle distributions.

The abundance of items in the simulated environment made the greedy search approach particularly effective, as the robot spent most of its time collecting or delivering rather than searching for items. This efficiency aligns with scenarios where targets are plentiful and replenished. However, the system does not balance collection across item types, which could be a limitation in real-world applications requiring proportional retrieval, such as order fulfilment. Implementing selection logic based on target ratios could address this.

3.3 Complexity

The solution's greedy search strategy directs the robot to the nearest detected item, identified by the largest apparent diameter in the camera view. This approach is computationally efficient and well-suited to small, item-rich environments like the simulated task. Its scalability is limited in larger or sparser environments, such as fulfilment centres, where the absence of global optimisation and item tracking would cause redundant travel and inefficient coverage. Without a memory of detected but uncollected items, the robot risks revisiting the same areas and failing to minimise travel distance. In such contexts, efficiency would require global mapping, item tracking, and optimised task allocation and multiple robots.

4 Safety and ethics

A key ethical consideration in deploying autonomous robots for this task is the potential implication for job displacement and wage reduction. A comparable real-world application is item transport in logistics and fulfilment centres, where companies such as Ocado and Amazon already employ and are expanding automation [3] [4]. Acemoglu et al. [5] estimate that introducing one robot per thousand workers can reduce the employment-to-population ratio by 0.2 percentage points and average wages by 0.42%. In large-scale warehouse operations, this could translate to significant reductions in income and employment opportunities for low-skilled workers. Ethical considerations therefore include how to mitigate these effects, such as providing retraining programs for new roles like robot supervisors, maintenance technicians, or system operators - however these strategies still may not suffice in addressing job displacement.

For this solution specifically, a major safety concern in real-world deployment is safe human–robot interaction. In settings such as hospitals or libraries for resource transport, the robot would certainly operate in shared spaces with humans. The current implementation relies solely on a 2D LiDAR sensor for obstacle detection and doesn't implement human detection. While effective for general navigation, it may fail to detect certain nuanced hazards, including stair edges or obstacles below scanner height. The onboard RGB camera lacks depth perception, reducing accuracy in estimating distances to humans or objects, particularly at greater ranges. Safety could be enhanced by integrating human detection algorithms, depth-capable cameras (e.g., stereo), and additional fail-safe behaviours.

The presence of a camera also raises privacy concerns, particularly regarding whether recorded imagery could be stored and repurposed, for example, to train deep learning control models. In sensitive environments such as hospital wards, robust data protection measures, including anonymisation, restricted storage, and compliance with relevant privacy regulations, would be essential to ensure ethical deployment.

5 Simulation Scenarios

Several simulation scenarios were developed to evaluate the robustness of the proposed solution under varying environmental conditions and scenarios.

In the first scenario, a delivery zone was removed to assess the robot's adaptability to changes in the operational environment. This reflects realistic situations in which certain delivery points may become inaccessible due to operational constraints or maintenance activities.

The second scenario involved reducing the number of available zones below the minimum required for task completion. For example, if a delivery truck departs a shipping depot, the robot should recognise that its assigned task is no longer feasible, halt operations, and notify supervisory personnel.

To conclude, while the solution could be enhanced through the integration of additional robots, improved navigation planning, and item tracking capabilities, overall, the robot remains effective at completing the task safely and reliably, and can generalise well to different scenarios due to its appropriate use of sensors and navigation systems.

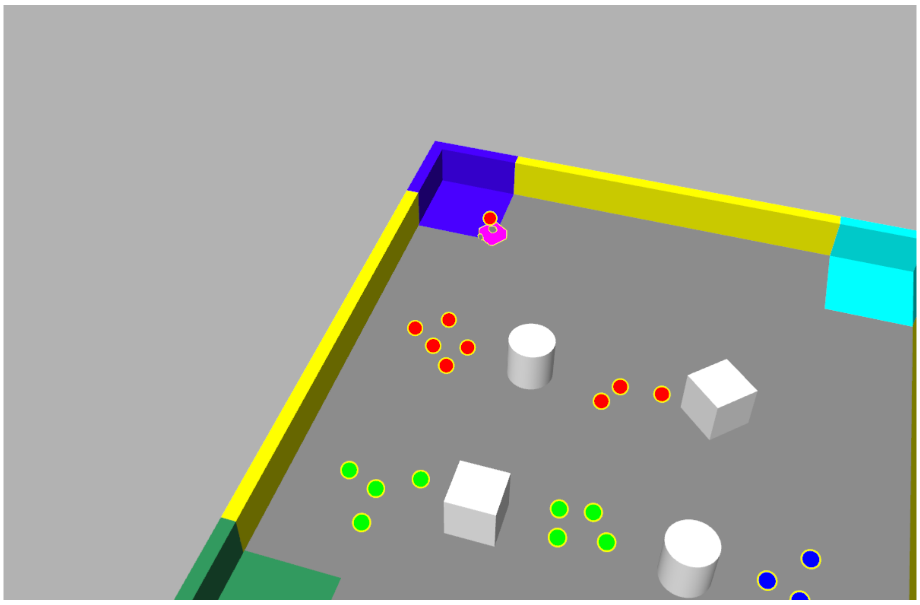

Figure 4: Gazebo Simulation Screenshot

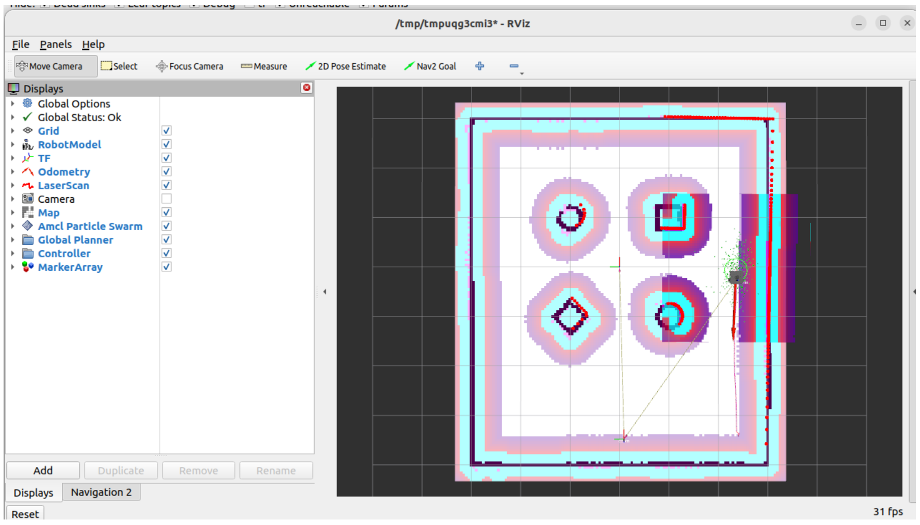

Figure 5: Rviz Simulation Screenshot

References

[1] A. Millard and P. Ribeiro, “robot_controller,” https://github.com/UoY-RoboStar/AURO/blob/main/week_5/week_5/robot_controller.py, 2024, accessed: Aug. 13, 2025.

[2] J. Newans, “ball_tracker,” https://github.com/joshnewans/ball_tracker, 2025, accessed: Aug. 13, 2025.

[3] D. Preston, “Inside the automated warehouse where robots are packing your groceries,” https://www.theverge.com/robot/719880/ocado-online-grocery-automation-krogers-luton-ogrp-robot-grid, 2025, accessed: Aug. 13, 2025.

[4] S. Herrera, “Amazon is on the cusp of using more robots than humans in its warehouses,” https://www.wsj.com/tech/amazon-warehouse-robots-automation-942b814f, 2025, accessed: Aug. 13, 2025.

[5] D. Acemoglu and P. Restrepo, “Robots and jobs: Evidence from us labor markets,” *Journal of Political Economy*, vol. 128, no. 6, pp. 2188–2244, 2020.