AMR Final Competition Report

Ivan Huang (qh229)

1. **Overview of the Team's Approach**

For the competition, our team implemented a system combining a **Particle Filter (PF)** for localization and **RRT with Dijkstra** for navigation and path planning. The robot is started at an unknown waypoint with arbitrary orientation and must localize, map optional walls, and visit all waypoints and EC waypoints while avoiding stay-away points.

The localization phase begins with initializing **700 particles at each possible waypoint** using initialPF. The robot then performs a **360-degree turn to gather depth and odometry data**, which initializes the PF and rapidly converges to a highly accurate robot pose estimate. This estimated pose is then used to determine the starting waypoint and store it as visited.

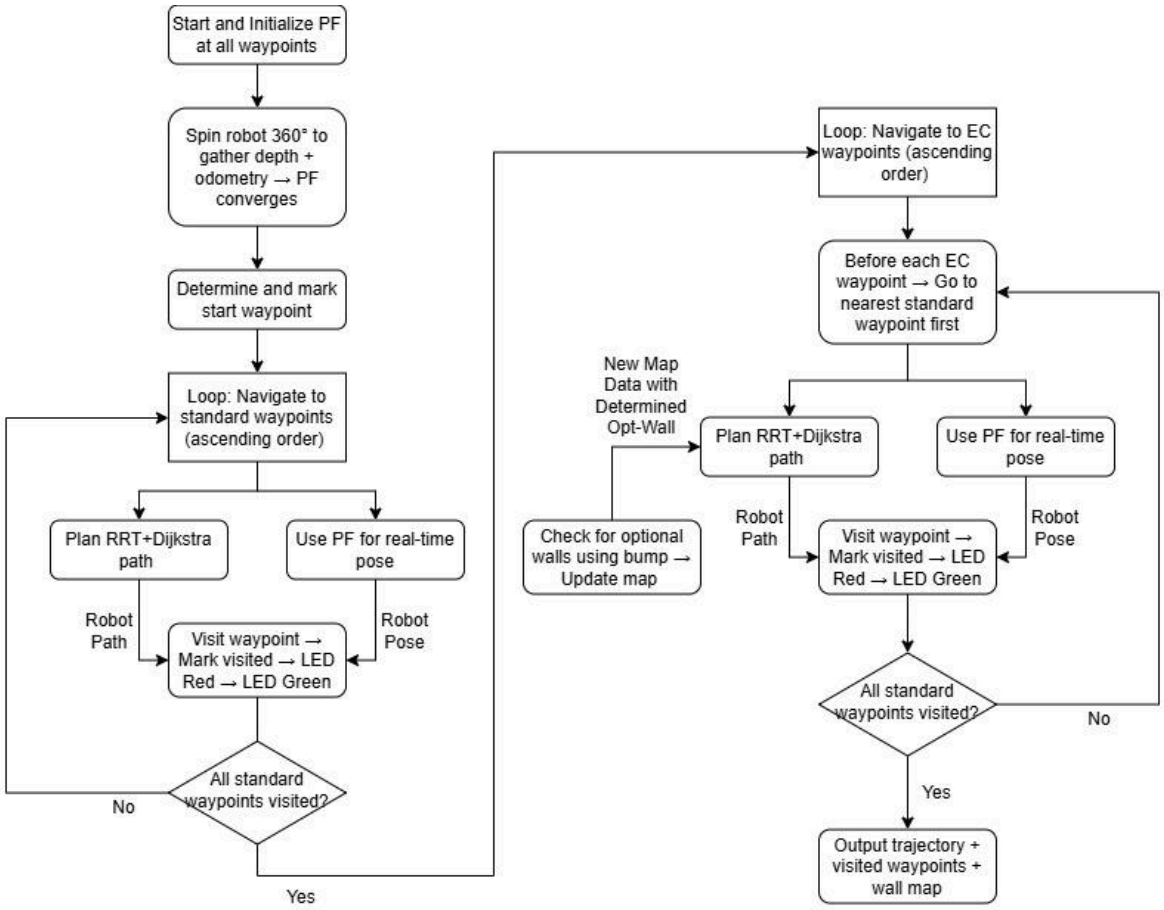
For navigation, the robot visits waypoints in ascending order using the navigPF function, which plans paths using **RRT** and selects the shortest path using **Dijkstra's algorithm**. Stay-away points are added into the map (inflated to a diamond shape) to be treated as obstacles during path planning. The robot follows the RRT path while updating its pose with the PF.

Once all standard waypoints are visited, the robot moves on to EC waypoints. To improve reliability, the robot first navigates to a nearby standard waypoint before targeting each EC waypoint. If the robot hits an optional wall during EC navigation (detected via bump sensors), the optional wall is marked as present and added into the map (goToWalls function). This dynamically updates the obstacle map and impacts future path planning.

While the approach worked very well in simulation with 3x speed, the limited robot speed during the real competition (0.1 m/s) and the tight 5-minute time limit limited the robot's ability to visit enough waypoints.

---

2. Flow Chart of Solution



---

3. Individual Contribution

I proposed and developed the entire approach, including:

- The use of **Particle Filter for localization** and **RRT with Dijkstra for navigation**.
- Designing the strategy to perform **initial localization with 700 particles and 360 degree spin**.
- Building and tuning the **Particle Filter for both initialization and navigation tracking**.
- Developing the navigPF and goToWalls functions, including:
  - Real-time integration of depth/odom/bump into PF.
  - RRT planning and waypoint visiting logic.
  - Collision handling and optional wall mapping logic.
- Designing the logic to visit **standard waypoints first**, and then EC waypoints for scoring.
- Implementing real-time plotting for debugging including:
  - Roadmap plotting
  - PF particles, pose, trajectory
  - Optional walls, stay-away zones, visited waypoints

- Integrating all modules and building the **top-level control logic** to manage task flow.
- Performing most of the testing and debugging:
  - Tuning PF parameters
  - Testing robot motion and waypoint visiting reliability
  - Identifying issues with speed, step size, waypoint visit distance

I spent significant time on integration and testing (estimate: ~25+ hours), iteratively improving reliability and robustness.

---

4. Discussion of Competition Performance

What worked well:

- **Localization using Particle Filter was highly accurate and stable**:
  During the competition, the robot consistently maintained accurate pose estimates. While running at the competition's required maximum speed of 0.1 m/s, the PF estimated pose closely tracked the true trajectory with a maximum deviation of ±0.1 meters. Both turning and straight movements were precisely captured in the PF estimate. This high accuracy enabled the robot to plan paths and make movement decisions reliably during operation.

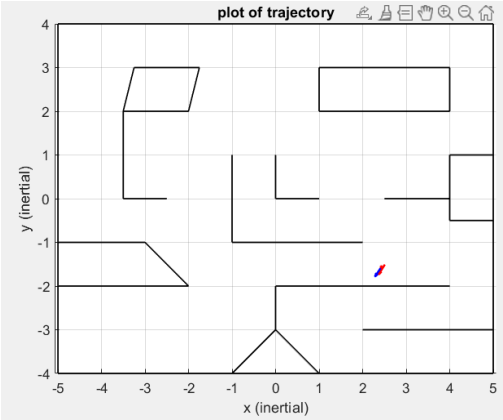


Figure 1. Initial PF localization

- **Path planning and waypoint navigation worked reliably, but conservatively**:
  The RRT and Dijkstra-based global planner generated valid paths that respected stay-away zones and obstacles. The robot followed these paths reasonably well at low speeds. However, the planner's small step size produced slightly jagged paths, which combined with slow speed, resulted in conservative and slow movement along the path. Still, the robot always made steady progress toward goals and avoided collisions.

- **EC waypoint routing logic was effective in simulation**:
  Although we did not reach EC waypoints during the competition (due to time constraints),

pre-competition testing demonstrated that our method of routing the robot through the nearest standard waypoint before targeting an EC waypoint significantly improved path planning success. Direct paths to EC waypoints often failed in tight spaces, while indirect paths via standard waypoints were consistently successful.

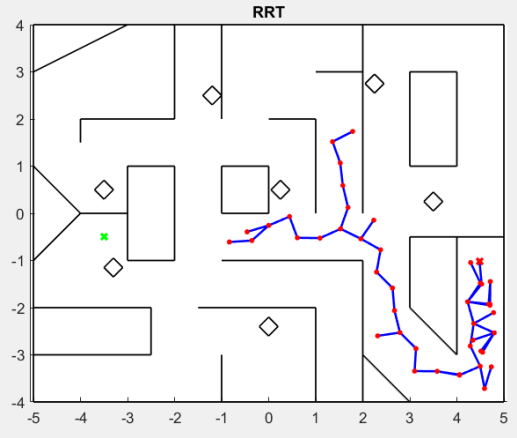


Figure . Direct paths to ECwp RRT Fail

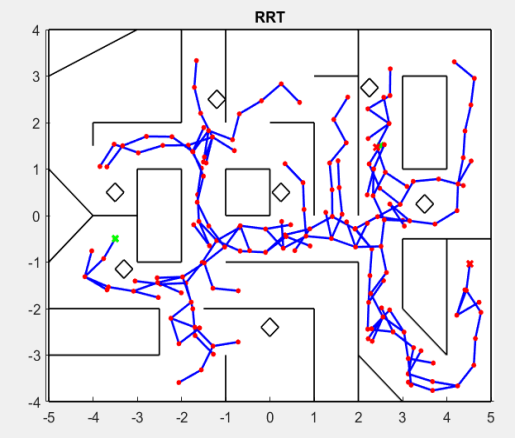


Figure . Indirect paths to ECwp RRT Success

- **System robustness and reliability**:
  The robot did not get stuck or encounter critical failures during competition. Even in the limited run time, it continuously made forward progress, automatically transitioning between waypoint targets. When the 5-minute timer expired, the robot was still actively navigating toward the next waypoint without being stalled or confused.

- **Visualization and debugging tools were invaluable**:
  The real-time plotting utilities we built (showing robot trajectory, PF pose, waypoints, optional wall statuses, and RRT roadmap) played a crucial role during testing and competition preparation. They allowed rapid tuning of PF parameters, adjustment of RRT step size, and validation of robot behaviors. The same visualization also served as the final map visualization required by the competition guidelines.

What did not work well and why:

- **Robot speed and competition time limit severely limited performance**:
  Our approach worked very well in simulation when running at 3x speed. The robot successfully visited all standard waypoints and even reached EC waypoints within 5 minutes. However, the real competition enforced a strict maximum speed of 0.1 m/s. At this speed, the robot could only reach its starting waypoint and one additional waypoint before time ran out. The bottleneck was not system correctness but time efficiency.

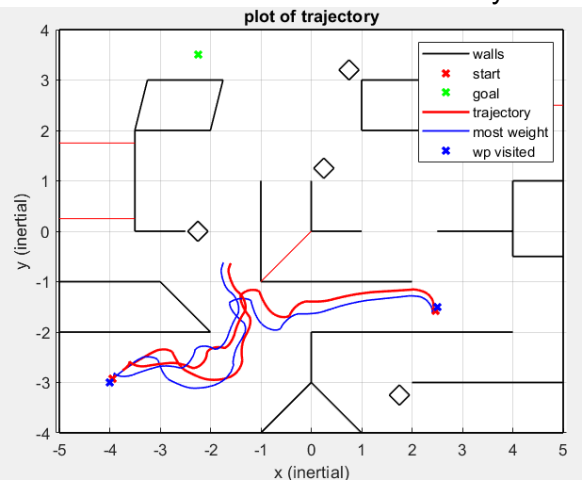


Figure . Robot Competition Simulation

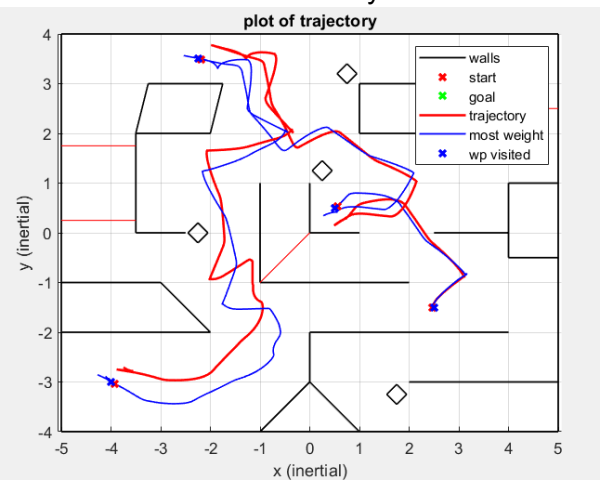


Figure . Robot Simulation With More Time

- **Path following was too conservative for competition time constraints**:
  The combination of:
  - Small RRT step size (resulting in jagged paths)
  - Strict “closeEnough” thresholds in waypoint visiting logic
  - Low robot speed limit

  resulted in slow and cautious path following. In hindsight, we realized that increasing the RRT step size and relaxing the visit distance threshold would have smoothed the robot’s movement and increased the number of waypoints reached. However, we only identified this improvement opportunity after the competition ended.

- **Optional wall detection and EC waypoint logic were not triggered due to time constraints**:
  Our approach for optional wall detection was based on bump sensing during EC waypoint navigation. Since the robot did not have enough time to visit EC waypoints during the competition run, this mechanism was not activated. However, even in testing, we observed a potential weakness with this method: requiring physical contact with walls not only risks robot mislocalization (due to bumps disturbing the pose estimate) but also is inefficient, as it requires very close proximity to the wall. In hindsight, using the depth sensor to detect optional walls at a distance would have been a superior method. Depth sensing allows detection earlier and at a safer distance without risking collisions, while still providing enough data to confirm whether a wall is present or absent. This is a key improvement we would implement in future versions.

Summary:

Overall, our system design proved robust, reliable, and technically sound. Localization was consistently accurate, planning and navigation were reliable, and our modular control structure performed well. The primary shortfall during the competition was not functional correctness but rather execution efficiency. The combination of slow maximum speed, cautious path planning, and tight time limit prevented the robot from reaching enough waypoints to score higher. Given more time or higher speed allowances, we are confident the robot could have performed much closer to its simulated results. For future improvements, tuning the RRT step size and waypoint proximity thresholds would significantly improve performance under time pressure.