Navigation & Localization Module Documentation

Package: `mam_eurobot_2026`

RoboUN Team - Eurobot 2026

January 5, 2026

Contents

1 Overview

The navigation stack enables the robot to move autonomously in the arena, creating a map of the environment and localizing itself within it. It is hosted within the `mam_eurobot_2026` package and integrates standard ROS 2 tools like **Nav2** and **SLAM Toolbox**, configured specifically for Eurobot rules.

Key Features:

- **Mapping:** Builds a 2D occupancy grid of the arena using LiDAR data.
- **Localization:** Estimates robot position (x, y, θ) using AMCL.
- **Path Planning:** Computes collision-free paths avoiding static and dynamic obstacles.
- **Motion Control:** Executes smooth trajectories via the differential drive controller.

2 Hardware Interfaces & Sensors

Unlike the Perception module which relies on cameras, the Navigation module depends on range and inertial sensors defined in the robot's SDF model.

2.1 2D LiDAR (RPLidar / Simulated)

- **Topic:** `/scan` (`sensor_msgs/LaserScan`)
- **Role:**
  - **Mapping:** Detecting static walls/obstacles to build `mymap.pgm`.
  - **Localization:** Matching current laser scans against the known map to correct odometry drift.
  - **Costmaps:** Updating the local costmap in real-time to avoid dynamic obstacles (e.g., opponent robot).

2.2 IMU / Gyroscope

- **Topic:** `/imu` (merged into `/odom`)
- **Role:** Provides rotational velocity data to determine the robot's **real rotation**. It compensates for wheel slip during fast turns to maintain an accurate heading (θ), critical for maneuvers like `half_spin.py`.

3 Algorithms & Logic

3.1 SLAM (Mapping Strategy)

The project uses **SLAM Toolbox** in *asynchronous online* mode to generate the static map.

- **Algorithm:** Graph-based SLAM (Karto).
- **Output:** An occupancy grid (`.pgm`) where:
  - Black pixels = Obstacles (Walls, Beacons).
  - White pixels = Free space.
  - Grey pixels = Unknown.

3.2 Localization: AMCL

During the match, the robot uses **Adaptive Monte Carlo Localization (AMCL)**.

1. **Particles:** Represents the robot's pose distribution as a cloud of particles.
2. **Prediction:** Moves particles based on Odometry (Wheels + IMU).
3. **Correction:** Weights particles based on how well the LiDAR scan matches the static Map.
4. **Resampling:** Converges the cloud around the true position.

TF Tree Role: AMCL publishes the correction transform `map` → `odom`.

4 Configuration & Tuning

The navigation behavior is strictly defined by YAML configuration files found in `mam_eurobot_2026/config/`.

4.1 SLAM Configuration

File: `config/slam/mapper_params_online_async.yaml`

- `resolution`: 0.05 m (Grid cell size)
- `max_laser_range`: 20.0 m
- `map_update_interval`: 5.0 s

4.2 Nav2 Configuration

File: `config/nav2/nav2_params.yaml`

- **Global Costmap:** Includes a Static Layer (Map) and an Inflation Layer (Safety radius).
- **Local Costmap:** A rolling window (3×3 m) centered on the robot, updated continuously by LiDAR.
- **Controller:** `DWBLocalPlanner` handles trajectory execution and velocity commands (`cmd_vel`).

5 Scripts & Behaviors

Custom Python scripts in `mam_eurobot_2026/lib/` implement specific game behaviors utilizing the navigation stack.

5.1 Navigation Commander (`auto_move.py`)

Acts as a basic Task Manager.

- Sends goal poses (x, y, θ) to the Nav2 Action Server.
- Monitors feedback and success status.
- Sequences movements between game elements (Start → Crate → Pantry).

5.2 Specific Maneuvers

- `half_spin.py`: Executes a precise 180° rotation using IMU feedback, useful for orientation calibration or scanning.
- `key_ctrl.py`: Provides manual teleoperation capabilities for testing or recovering the robot.

6 Usage Example

6.1 Launching the Simulation Stack

To start Gazebo, Robot Description, and the Navigation Stack simultaneously:

```
ros2 launch mam_eurobot_2026 arena.launch.py
```

6.2 Visualizing the Map

Open **RViz** to inspect the SLAM/AMCL status.

- **Map Topic:** `/map`
- **Particle Cloud:** `/particlecloud` (visualizes AMCL uncertainty)
- **Costmaps:** `/global_costmap/costmap`

6.3 Manual Override

If the robot is stuck, launch the teleop script:

```
ros2 run mam_eurobot_2026 key_ctrl.py
```