Perception Module Documentation

Package: `mam_eurobot_2026`

RoboUN Team - Eurobot 2026

January 5, 2026

Contents

1 Overview

The perception capabilities are hosted within the main project package, `mam_eurobot_2026`. The system utilizes a **Dual-Camera Architecture** combined with an advanced **ArUco-based Localization System**.

1.1 The Camera Setup

- **Eagle Eye Camera (Static):** A fixed camera mounted in the corner of the arena (`left_corner_cam`). It uses **Color Detection** for global inventory and can **Self-Calibrate** using reference ArUco tags on the field.
- **Robot Camera (Onboard):** A camera mounted on the robot (`simple_robot`). It uses **ArUco Tag Detection** for high-precision alignment and grasping.

Source File: `mam_eurobot_2026/open_cv_cam.py`

2 Capabilities & Roles

2.1 Eagle Eye Perception (Global Inventory)

- **Goal:** Provide a global map of where the blue and yellow crates are located.
- **Method: HSV Color Segmentation** (primary) and **Reference ArUco Calibration** (setup).
- **Output:** Publishes coordinates (X, Y) in the `map` frame.

2.2 Onboard Perception (Precision Docking)

- **Goal:** Guide the **gripper** to the exact center of the crate.
- **Method: ArUco Marker Tracking** (IDs 36, 47, 41).
- **Output:** Publishes precise 6-DOF pose (x, y, z, yaw) relative to the Robot Base or World Frame (if reference tags are visible).

3 Algorithms & Logic

3.1 Reference-Based Self-Localization (Code Logic)

The `CameraViewer` node implements a dynamic localization feature. Instead of relying solely on hardcoded parameters, it can compute the camera's absolute position in the world frame (T_{world_camera}) by detecting fixed reference markers on the arena floor/walls.

Algorithm:

1. Detect reference markers (IDs 20–23) in the image.
2. Estimate pose *rvec*, *tvec* for each marker $\rightarrow T_{camera \rightarrow marker}$.
3. Invert to get $T_{marker \rightarrow camera}$.
4. Compute camera pose:

$$T_{world \rightarrow camera} = T_{world \rightarrow marker} \times T_{marker \rightarrow camera}$$

5. Average the matrices if multiple markers are seen to get a robust `T_world_camera_final`.

3.2 Crate Localization (Chained Transform)

Once the camera pose T_{world_camera} is known (either computed dynamically or loaded from static parameters), crate positions are calculated in the global frame:

1. Detect crate marker (e.g., ID 36).
2. Compute $T_{camera \to crate}$ using `cv2.aruco.estimatePoseSingleMarkers`.
3. Compute Global Pose:

$$T_{world \to crate} = T_{world \to camera} \times T_{camera \to crate} \tag{1}$$

3.3 Global Color Detection (Fallback/Eagle)

For objects without tags or when reference tags are occluded:

- **Color Filtering (HSV):**
  - Yellow Crates: $H \in [25, 35]$
  - Blue Crates: $H \in [100, 120]$
- **Global Transformation (Extrinsics):** Uses the pre-calibrated T_{world_camera} to project pixel centroids onto the map plane ($Z = 0$).

3.4 System Integration (Handoff Strategy)

- **Approach:** Robot moves to the Global coordinates.
- **Switch:** When distance to target < 10cm, switch to Onboard ArUco tracking.

4 Node API: `open_cv_cam`

Class: `CameraViewer`

4.1 Subscribed Topics

Topic	Message Type	Description
`/camera/image_raw`	`sensor_msgs/Image`	Raw RGB video stream.
`/camera/camera_info`	`sensor_msgs/CameraInfo`	Camera intrinsics (K, D).

4.2 Published Topics

Topic	Message Type	Content
`detections`	`eurobot_interfaces/CrateDetectionArray`	List of detected objects/poses.
`debug_image`	`sensor_msgs/Image`	Visual overlay with axes.

4.3 Parameters & Constants (Internal)

The following are defined in the `__init__` method of the node:

- `marker_length` (Reference): **0.1 m** (100 mm)

- `crate_marker_length`: **0.0375 m** (37.5 mm)
- `aruco_dict`: `DICT_4X4_1000`
- `debug_camera`: Boolean to print camera pose transforms.
- `debug_crates`: Boolean to print crate pose transforms.

5 Hardware Specifications

5.1 Onboard Camera (`simple_robot`)

- **Mounting:** $x = 0.085$, $z = 0.04$ (relative to chassis).
- **Tilt (Pitch):** 0.5236 rad (30°).
- **Resolution:** 640×480.

5.2 Eagle Eye Camera (`left_corner_cam`)

- **World Position (x, y, z):** $1.950, -1.550, 1.0$ meters.
- **Orientation:** Pitch $\approx 30°$, Yaw $\approx 143°$.

5.3 Tags & Objects

- **Crate Tag Size:** 37.5 mm (ArUco ID 36, 47, 41).
- **Reference Tag Size:** 100 mm (ArUco ID 20, 21, 22, 23).
- **Crate Dimensions:** $150 \times 50 \times 30$ mm.

6 Usage Example

To launch the node:

```
ros2 run mam_eurobot_2026 open_cv_cam
```

Note: Ensure reference tags are visible in the camera frame for the first few frames to initialize `T_world_camera_final`.