# Test and Validation Plan

*Project: Viscon Operational Robotics Engineer Assignment 10 Jun 2025*

## Overview & objectives

This document outlines the testing and validation approach for the ROS2 nodes developed to detect and count rails in greenhouse environments using the EVA mobile robot. The objectives of the test and validation are:

- Ensure reliable rail detection under diverse operational conditions using a real robot.
- Confirm accurate counting of rows as the robot navigates through greenhouse aisles.

## Equipment

- EVA Scoutr; Viscon Level 4 Autonomous Mobile Robot
- Intel RealSense D456 camera
- Pre-trained UNet rail detection model
- ROS2 environment and required software dependencies
- Odometry data capturing setup
- Laptop or PC equipped with ROS2 and visualization tools (e.g., RQT, RViz)

## Requirements and acceptance criteria

### *Functional requirements*

- Rail detection must operate in real-time, providing results at the camera frame rate (5 Hz).
- Row counting must provide an integer output corresponding to the actual rows passed.
- Both nodes must integrate within the existing ROS2 framework on the robot.

### *Performance requirements*

- Detection accuracy must exceed XXX % precision and recall.
- Row counting accuracy must achieve at least ≤ ±1 error per XXX hour.

### *Environmental robustness requirements*

- Nodes must maintain accuracy under variable lighting conditions (natural daylight, artificial lighting, nighttime).
- Performance stability under diverse greenhouse environmental conditions (temperature, humidity, condensation).

*Acceptance criteria*

- Nodes consistently meet or exceed specified accuracy metrics.
- Demonstrable stability over extended operational durations (22/7 continuous operation without failure).
- System successfully identifies and logs rail locations and counts with minimal manual intervention.

## System overview

| Node | Inputs | Outputs | Parameters |
|---|---|---|---|
| rail_detector_node | /d456_pole/infra1/image_rect_raw (1280×720 @ 5 Hz) | /rail_mask /rail_detector/overlay | Min_area |
| row_counter_node | /rail_mask, /odometry/filtered | /rows_count | row_spacing, min_area_px, min_overlap_px |

## Testing environment

The testing will take place in the operational greenhouse, replicating real-world conditions, including:

- Variable lighting conditions (natural day-night cycles and artificial lighting)
- Different environmental conditions (humidity, condensation, temperature)
- Typical obstacles and reflective surfaces (metal heating pipes, glass panels, water droplets, plants)

## Data Logging

- Detections, timestamped with odometry information
- Environmental metadata (lighting condition, humidity, temperature)
- Explicit logging of anomalies and deviations.

## Validation metrics

- **Detection accuracy**: E.g., precision, recall, IoU.
- **Counting accuracy**: Actual vs. counted rows (verified manually)

## Reporting

- Periodic reports summarising accuracy and detection performance.
- Final report including recommendations for system improvements.

# Testing procedure

## 1. Initial functional testing

*Objective:* Confirm basic functionality, integration, and real-time performance.

*Procedure*

1. Set up the robot with all components (camera, ROS nodes, visualization tools).
2. Launch the nodes using the provided rail_detector_launch.py script.
3. Confirm the nodes start correctly by checking the ROS topic subscriptions and publications (ros2 topic list).
4. Manually control the robot to traverse one aisle slowly.
5. Monitor real-time detection outputs visually using rqt_image_view.
6. Record detected rails and confirm detections correspond to actual rails.

*Data Logging:*

- ROS bag recording of all relevant ROS topics.
- Log initial test outcomes and node status checks.

*Validation:*

- Review the recorded ROS bag data offline with RViz/rqt to check detections.
- Manually verify image detections against physical locations of rails.
- Compare the actual number of rows to counted number of rows.

## 2. Extended operational testing

*Objective:* Assess detection accuracy across various lighting conditions (daytime, twilight, nighttime) and ensure robustness during continuous 22/7 operation.

*Procedure:*

1. Capture additional image data in varying operational conditions.
2. Preprocess images using scripts (preprocess_images.py and postprocess_images.py).
3. Run detection model on preprocessed images.
4. Configure robot for extended autonomous operation.
5. Schedule test runs across distinct lighting conditions (morning, noon, evening, night).
6. Begin continuous test for 48 hours, ensuring periodic monitoring.
7. Regularly check robot status, ROS node health, and logged data for anomalies.
8. Record environmental conditions (e.g., time, sunlight intensity, artificial lighting status).

*Data Logging:*

- Continuous ROS bag recording in segments of X-hour duration.
- Maintain a logbook noting any environmental changes and observations.

*Validation:*

- Post-operation analysis of ROS bag files, comparing detection continuity and consistency.
- Identify periods of failure or inconsistency and correlate them with environmental notes.

## Known challenges and mitigation strategies

- **Rail detection dependency:** Row counting directly relies on successful rail detection. Regular checks should be implemented to identify rail detection failures.
- **Intermittent rail visibility**: Partial or temporary visibility loss (due to reflections, obstructions, or environmental factors) could impair counting accuracy. Mitigate by ensuring robust preprocessing (CLAHE, Gaussian blur) and by integrating fallback logic, e.g., using historical odometry data to maintain row counting integrity temporarily.
- **Variable lighting conditions:** Regularly update training datasets and include images captured across multiple lighting scenarios. Implement adaptive preprocessing techniques to handle diverse lighting conditions dynamically.
- **Environmental variations:** Regular validation checks, dynamic parameter tuning for thresholding and model adjustments.

## Conclusion

Following this detailed test and validation plan ensures reliable performance, robust rail detection, and accurate row counting in actual greenhouse operational conditions.