System Verification and Validation Plan for 2D Localizer

Aliyah Jimoh

March 3, 2025

Revision History

Date	Version	Notes
24/02/2025	1.0	Initial Draft
03/03/2025	1.1	Feedback Implementation

Contents

List of Tables

List of Figures

[Remove this section if it isn't needed —SS]

1 Symbols, Abbreviations, and Acronyms

Symbol	Description
2D	Two-Dimensional
2D Localizer	2D Localization Solution
CRLB	Cramér-Rao Lower Bound
FM	Fiducial Marker
GTSAM	Georgia Tech Smoothing and Mapping
MG	Module Guide
MIS	Module Interface Specification
MLE	Maximum Likelihood Estimation
PEP8	Python Enhancement Proposal 8
SE(2)	Special Euclidean Group in 2D
SRS	Software Requirements Specification
T	Test
VnV	Verification and Validation

For a full list of symbols, abbreviations, and acronyms used, refer to section 1 in the SRS document.

This document shows the verification and validation plan of the 2D Localizer program. This plan starts with general information that talks about 2D Localizer in section 2. The plan and system tests involved with this software is explained in sections 3 and 4.

2 General Information

2.1 Summary

This document examines the verification and validation plan of 2D Localizer. This software is used to help accurately locate mobile robots in a provided 2D map given the measurements and coordinates of the sensors and fiducial markers (FMs).

2.2 Objectives

The objective of this plan is to validate the accuracy of this program to build confidence in the software correctness. This plan also aims to satisfy all the requirements the System Requirements Specification (SRS) document outlined.

2.3 Challenge Level and Extras

This system has an advanced research level which can be seen from the implementation and the topic. Setting up the robot's movement, accurately displaying the trajectory, coordinating the sensors' measurements, and finding a way to animate the output would definitely add difficulty to this system, however, this is not a niche topic meaning that there are papers or libraries available to draw inspiration from.

2.4 Relevant Documentation

The documentation relevant to the 2D Localizer includes the Problem Statement since it explains the proposed software, the SRS which talks about the requirements needed to properly use the system, the Verification and Validation (VnV) report that goes through the tests and plans for the system, and the Module Guide (MG) and Module Interface Specification (MIS) for the design.

3 Plan

This section describes the tests made for the 2D Localizer system. This begins by mentioning the VnV team in section 3.1, then followed by the SRS Verification Plan in 3.2, the Design Verification Plan in section 3.3,the VnV Plan Verification Plan in section 3.4, the Implementation Verification Plan in section 3.5, the Automated Testing and Verification tools in section 3.6, and the Software Validation Plan in section 3.7.

3.1 Verification and Validation Team

This section shows the members of the VnV team. They are shown in Table 1 along with what document they contributed in and their roles.

Name	**Document**	**Role**
Aliyah Jimoh	All	Author
Dr. Spencer Smith	All	Instructor Reviewer
Kiran Singh	SRS VnV	Domain Expert Reviewer
Dr. Matthew Giamou	Problem Statement	Supervisor Reviewer

Table 1: Verification and Validation Team

3.2 SRS Verification Plan

The SRS document for 2D Localizer will be verified through the following steps:

1. The initial review will be preformed by the assigned reviewers (Dr. Spencer Smith and Kiran Singh) and will use the SRS Checklist as a guide.

2. The reviewers will give feedback through creating issues in the GitHub repository.

3. The author (Aliyah Jimoh) will apply the feedback to the document and address issues that may not be applied.

4. The author will discuss with the supervisor (Dr. Matthew Giamou) and ask for their input.

3.3 Design Verification Plan

The design for 2D Localizer will be verified through the following steps:

1. The design documents, the MG and MIS, will be reviewed by the assigned reviewers (Kiran Singh and Dr. Spencer Smith) and will use both checklists (MG Checklist and MIS Checklist) as a guide.

2. The reviewers will give feedback through creating issues in the GitHub repository.

3. The author (Aliyah Jimoh) will apply the feedback to the document and address issues that may not be applied.

3.4 Verification and Validation Plan Verification Plan

The VnV plan for 2D Localizer will be verified through the following steps:

1. The VnV plan will be reviewed by the assigned reviewers (Kiran Singh and Dr. Spencer Smith) and will use the VnV Checklist as a guide.

2. The reviewers will give feedback through creating issues in the GitHub repository.

3. The author (Aliyah Jimoh) will apply the feedback to the document and address issues that may not be applied.

3.5 Implementation Verification Plan

The implementation of 2D Localizer will be verified be the following techniques:

- **Static Verification**
  The author will perform a code walkthrough the domain expert (Kiran Singh) and go through different tests to check the accuracy. The author

will then do a walkthrough in a presentation while explaining the tests made and performed.

- **Dynamic Testing**
  The unit tests for 2D Localizer are discussed in section 5

3.6 Automated Testing and Verification Tools

3.6.1 Code Guidelines

For Python implementation, the 2D Localization software will follow the coding style guidelines outlined in Python Enhancement Proposal 8 (PEP8). To enforce this, the Flake8 linter will be used to ensure that code meets PEP8's guidelines and for continuous integration in GitHub Actions.

3.6.2 GTSAM Integration

Georgia Tech Smoothing and Mapping (GTSAM) by Dellaert and Contributors (2022) is a library that implements sensor fusion for robotics using factor graphs. These graphs consist of two types of nodes: variables (e.g., robot poses and sensor measurements) and factors, which represent probabilistic constraints between them. 2D Localizer will be using this library as a modelling language and would need to run unit tests to verify if GTSAM is properly integrated.

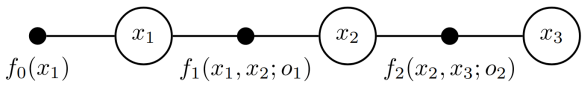

Figure 1: Example Factor Graph from Dellaert (2012)

3.7 Software Validation Plan

To help validate the accuracy of the localization algorithm, 2D Localizer will use 2D data sets to test. If a data set cannot be found, alternative test cases and feedback will be discussed with the supervisor.

4 System Tests

This section goes over the different system tests 2D Localizer will go through to satisfy its requirements.

4.1 Tests for Functional Requirements

This subsection looks at the system tests that will help satisfy the functional requirements outlined in the SRS (Section 5.1)

4.1.1 Sensor Estimation Tests

These tests cover the requirements R1 to R3

Range-Only Pose Estimation

1. F-RO-01

   Control: Automatic

   Initial State:

   - Known beacon positions
   - Robot starts at unknown location

   Input: Range measurements $\tilde{d}_j$ from beacons with noise

   Output: The estimated robot pose $\hat{x}$, as defined in the SRS (Section 5.1, Requirement R3).

   Test Case Derivation: SRS Section IM2

   How test will be performed:

   - Use 2D dataset
   - Run GTSAM-based factor graph optimization to estimate $\hat{x}$
   - Verify the estimated error bounds align with FIM-derived uncertainty

SE(2) Transformation Estimation

1. F-SE2-01

   Control: Automatic

   Initial State:

   - Robot starts with a known pose T_{wf}
   - Fiducial markers are placed at fixed positions

   Input: Transformation matrices T_{wf}, T_{br}

   Output: Computed transformation must satisfy:

   $$T_{wr} = T_{wf} T_{fr}$$

   Test Case Derivation: The transformation matrix T_{wr} is computed using the rigid-body transformation rule in SE(2), which is required to maintain the system's geometric consistency (SRS Section TM3)

   How test will be performed:

   - Compute T_{wr} using both analytical and numerical methods
   - Compare GTSAM-computed pose with theoretical SE(2) output

Range + SE(2) Sensor Fusion Test

1. F-SF-01

   Control: Automatic

   Initial State:

   - Robot starts at unknown pose
   - Beacons and fiducial markers placed in test environment

   Input: Range measurements + SE(2) transformation constraints

   Output: Estimated pose T_{wr} and $\hat{x}$

   Test Case Derivation: F-RO-01 and F-SE2-01

   How test will be performed:

- Construct factor graph in GTSAM
- Run optimization twice (once with range only, once with full fusion)
- Compare error reductions

4.1.2 Visual Testing

This test covers the requirement R5

2D Map Overlay of Estimated Poses

1. F-MO-01

   Control: Automatic

   Initial State:

   - Environment 2D map and sensor locations (beacons, fiducials) are known.
   - Robot follows a predefined trajectory.

   Input:

   - Ground truth trajectory T_{wr}
   - Estimated trajectory from localization output $\hat{x}$, generated by a GTSAM-based factor graph optimization.
   - Sensor positions and detections.

   Output: The visualization correctly displays:

   - Estimated trajectory $\hat{x}$ based on GTSAM's factor graph optimization.
   - Sensor placements (beacons, fiducial markers).

   Test Case Derivation: The estimated pose $\hat{x}$ is computed using Maximum Likelihood Estimation (MLE), as described in SRS Section IM2.

   How test will be performed:

   - Run GTSAM-based localization and extract optimized poses $\hat{x}$.

- Generate a visualization overlaying $\hat{x}$ on the environment map.
- Compare the generated visualization with an expected ground-truth image.

4.2 Tests for Nonfunctional Requirements

4.2.1 Estimation Accuracy

Accuracy Validation

1. NF-ACC-01

   Type: Nonfunctional, Dynamic, Automatic

   Initial State:

   - The system has access to ground truth robot poses $\bar{x}$.
   - Sensor measurements (range, fiducial marker detections) are available.

   Input/Condition:

   - Synthetic and real-world sensor data.
   - Estimated robot poses $\hat{x}$, generated using a GTSAM-based factor graph optimization.
   - Ground truth robot trajectory.

   Output/Result:

   - The Root Mean Squared Error (RMSE) of the estimated poses $\hat{x}$ must satisfy the accuracy constraint in SRS Section 5.2, Requirement NFR1.
   - The estimated pose covariance must not exceed the theoretical Cramér-Rao Lower Bound (CRLB) (SRS Section TM2).

   How test will be performed:

   - Generate test cases with ground-truth trajectory $\bar{x}$.
   - Compute estimated poses $\hat{x}$ using GTSAM factor graph optimization.

- Evaluate RMSE between $\hat{x}$ and $\bar{x}$:
- Compare estimated variance to the CRLB (TM3)

4.3 Traceability Between Test Cases and Requirements

	F-RO-01	F-SE2-01	F-SF-01	F-MO-01	NF-ACC-01
R1	X	X	X	X	
R2	X		X		X
R3	X		X		
R4					
R5				X	

Table 2: Traceability Matrix Showing the Connections Between Requirements and System Tests

5 Unit Test Description

[This section should not be filled in until after the MIS (detailed design document) has been completed. —SS]

[Reference your MIS (detailed design document) and explain your overall philosophy for test case selection. —SS]

[To save space and time, it may be an option to provide less detail in this section. For the unit tests you can potentially layout your testing strategy here. That is, you can explain how tests will be selected for each module. For instance, your test building approach could be test cases for each access program, including one test for normal behaviour and as many tests as needed for edge cases. Rather than create the details of the input and output here, you could point to the unit testing code. For this to work, you code needs to be well-documented, with meaningful names for all of the tests. —SS]

5.1 Unit Testing Scope

[What modules are outside of the scope. If there are modules that are developed by someone else, then you would say here if you aren't planning on verifying them. There may also be modules that are part of your software, but have a lower priority for verification than others. If this is the case, explain your rationale for the ranking of module importance. —SS]

5.2 Tests for Functional Requirements

[Most of the verification will be through automated unit testing. If appropriate specific modules can be verified by a non-testing based technique. That can also be documented in this section. —SS]

5.2.1 Module 1

[Include a blurb here to explain why the subsections below cover the module. References to the MIS would be good. You will want tests from a black box perspective and from a white box perspective. Explain to the reader how the tests were selected. —SS]

1. test-id1

   Type: [Functional, Dynamic, Manual, Automatic, Static etc. Most will be automatic —SS]

   Initial State:

   Input:

   Output: [The expected result for the given inputs —SS]

   Test Case Derivation: [Justify the expected value given in the Output field —SS]

   How test will be performed:

2. test-id2

   Type: [Functional, Dynamic, Manual, Automatic, Static etc. Most will be automatic —SS]

   Initial State:

   Input:

   Output: [The expected result for the given inputs —SS]

   Test Case Derivation: [Justify the expected value given in the Output field —SS]

   How test will be performed:

3. ...

5.2.2 Module 2

...

5.3 Tests for Nonfunctional Requirements

[If there is a module that needs to be independently assessed for performance, those test cases can go here. In some projects, planning for nonfunctional tests of units will not be that relevant. —SS]

[These tests may involve collecting performance data from previously mentioned functional tests. —SS]

5.3.1 Module ?

1. test-id1

   Type: [Functional, Dynamic, Manual, Automatic, Static etc. Most will be automatic —SS]

   Initial State:

   Input/Condition:

   Output/Result:

   How test will be performed:

2. test-id2

   Type: Functional, Dynamic, Manual, Static etc.

   Initial State:

   Input:

   Output:

   How test will be performed:

5.3.2 Module ?

...

5.4 Traceability Between Test Cases and Modules

[Provide evidence that all of the modules have been considered. —SS]

References

Frank Dellaert. Factor graphs and gtsam: A hands-on introduction. 2012. URL https://api.semanticscholar.org/CorpusID:131215724.

Frank Dellaert and GTSAM Contributors. borglab/gtsam, May 2022. URL https://github.com/borglab/gtsam).

6 Appendix

This is where you can place additional information.

6.1 Symbolic Parameters

The definition of the test cases will call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance.

6.2 Usability Survey Questions?

[This is a section that would be appropriate for some projects. —SS]