Reflection and Traceability Report on 2D Localizer

Aliyah Jimoh

This reflection document goes over the changes made for 2D Localizer including the SRS, VnV Plan and Report, and design documents (MG and MIS).

1 Changes in Response to Feedback

1.1 SRS and Hazard Analysis

This table addresses the main feedback implemented to the SRS document under commit 28eaed8. Feedback was given from Instructor and Domain Expert (GitHub Issues)

Source	**Feedback**	**Response**
Instructor	Add missing units, define all symbols	Updated Table of Symbols with missing units and added map size constraints
Domain Expert (Issue #8)	Define pose and possibly a diagram for clarification	Refined definition of pose an added diagram to clarify SE(2) interpretation in Terminology section
Instructor	Inconsistent notation in Gaussian noise model	Rewrote Gaussian noise model sections to clearly define noise and distribution assumptions
Instructor	SE(2) transformation usage unclear	Clarified usage of SE(2) transformations in relevant models and added matrix explanation
Instructor	Original TM1 undefined in context of models	Revised or removed undefined theoretical models in favour of DD1
Domain Expert (Issue #19)	NFR4 lacks specification of standard libraries and OS compatibility	Updated NFR4 (Now NFR3) to include the operating systems and standard libraries the software should be compatible with
Domain Expert (Issue #12)	Input data constraint table missing uncertainties and physical meaning	Completed input and output tables with uncertainty, typical values, and physical constraints
Instructor	Non-Functional requirements are too ambiguous	Refined non-functional requirements for clarity and specificity
Domain Expert (Issue #14)	R5 should be broken down into more requirements	Broke down R5 to create requirements R5-R9 for more specific goals

Table 1: SRS and Hazard Analysis Feedback Response

1.2 Design and Design Documentation

This table addresses the main feedback implemented to the MG and MIS documents under commit 72a615f. Feedback was given from Instructor and Domain Expert (GitHub Issues)

Source	Feedback	Response
Instructor (MIS)	GUI dependency not mentioned	Updated environmental variables to show updated variables
Instructor (MIS)	GTSAM types not explained well in Notation section	Moved data types to the Exported Types section and refered to GTSAM API for more information
Instructor (MIS)	Exported programs poorly described	Added access routine descriptions, particularly for Control and Output modules
Domain Expert (Issue #34) - MG	Module 5 (Localization Module) addresses Anticipated Changes AC3 and AC5.	Reassessed and clarified the separation of concerns between AC3 and AC5 in Module 5
Domain Expert (Issue #36) - MG	Plotting and Output module roles overlapping	Distinction added: Plotting handles visualization; Output formats tabular results
Domain Expert (Issue #38) - MG	M5 and M8 (Accuracy Evaluation Module) module hierarchy unclear in Figure 1	Redid module hierarchy in Figure 1 to reflect appropriate levels for all modules
Domain Expert (Issue #40) - MIS	Plotting Module states trajectory is only called when a valid estimate exists, but there is no exception	Added an exception to the Localization module to verify if an estimate is valid before plotting
Domain Expert (Issue #43) - MIS	Notable number of variables are undefined with respect to their context	Added a comprehensive table defining all variables and their contexts within the MIS

Table 2: Design and Documentation Feedback Response

1.3 VnV Plan and Report

This table addresses the main feedback implemented to the VnV Plan document under commit 58bee25. Feedback was given from Instructor and Domain Expert (GitHub Issues)

Source	Feedback	Response
Domain Expert (Issue #22)	Section 2.2 lacks a clear definition of "correctness"	Added context in Section 2.2 to define "correctness" in terms of reliability and an accurate performance.
Domain Expert (Issue #24)	Characteristics of an acceptable dataset for validation activities are undefined	Defined the characteristics of acceptable datasets, including data format, range, and quality requirements for validation
Domain Expert (Issue #27)	Ambiguous definition of "2D dataset" and unclear if FIM-derived uncertainty is calculated or provided	Clarified that "2D dataset" refers to to trajectory and measurement files structured in the input CSV files.
Domain Expert (Issue #28)	Transformations like T_{wf} and T_{fr} are not defined or included in the Table of Symbols	Added definitions for T_{wf} and T_{fr} in the Table of Symbols and ensured consistent usage throughout the document
Instructor	Vague description of unit testing scope and coverage	Expanded Unit Testing Scope and clearly mapped each module to corresponding test files
Instructor	Requirements not being tested according to the Traceability Matrix	Fleshed out systems tests for all requirements to be met.

Table 3: VnV Plan and Report Feedback Response

The VnV Report will be initially released under the commit of the final documentation.

2 Challenge Level and Extras

2.1 Challenge Level

This project is considered as an advanced levelled research project.

2.2 Extras

There are no extra deliverables in this project.

3 Design Iteration (LO11 (PrototypeIterate))

The original design aimed to validate the accuracy of a 2D localization solution using predefined datasets that would have been referenced from a research paper. However, as the project progressed, several key implementation and usability challenges led to changes in the design.

First, the plan to rely on a single predefined dataset was reconsidered. Since the localization algorithm must work across varying map sizes and input trajectories, the system was restructured to accept multiple CSV files for different test cases. This allowed greater flexibility in experimentation and better reflected the potential variability of real-world scenarios.

To better understand the input structure and how localization behaves under different conditions, a Simulation Module was introduced. This module

generates trajectory and sensor readings directly within a known environment. This was not only useful for validation but also helped in refining test data, debugging estimation inconsistencies, and ensuring robustness across inputs.

Additionally, an Output Module was introduced to provide a tabular summary of pose estimates. Initially, the plan was to have the pose estimate only visible through the Plotting Module's live GUI display. However, this dependency made it difficult to assess results in a straightforward manner. By separating the output logging into its own module, the system now stores final pose estimates in a structured format (e.g., tables), which supports reproducibility, documentation, and accuracy evaluation.

4 Design Decisions (LO12)

This section reflects on the key design decisions that shaped the final implementation of 2D Localizer. These decisions were made in light of the project's limitations, assumptions, and constraints.

Limitations

A significant limitation was the scope of sensor data and environment complexity. The system operates in a 2D space and assumes that the robot operates on a flat plane. This limited the need for full SE(3) transformations, allowing SE(2)-based modeling instead. However, this helped with simplifying the calculations as it involved fewer components which is why discussions with the supervisor concluded that it would be 2D.

Another limitation was reliance on the PyPI build of GTSAM, which excluded certain features that could help simplify estimation solutions. This influenced the design to rely on supported factor types and to wrap only the minimal necessary subset of GTSAM functionality in a custom interface.

Assumptions

The system assumes that all sensor positions (beacons and fiducial markers) are known and remain static throughout operation. It also assumes that input files are correctly formatted and that range and transformation data follow the defined structure in the YAML and CSV files. These assumptions informed the design of the Input Format and Simulation modules, which enforce structure validation and data sanity checks.

It was also assumed that users would benefit from observing pose estimates both visually and numerically. This motivated the separation of plotting and output functionalities into distinct modules.

Constraints

One of the main constraints was ensuring modularity and traceability. Each module had to serve a specific role (e.g., localization, plotting, accuracy evalu-

ation) with well-documented inputs and outputs. This modular structure was necessary both for validation and for meeting assignment standards like traceability matrices and MIS documentation.

Time was another constraint, especially in developing reusable validation cases. As a result, automated testing with PyTest and flake8 was prioritized. Usability considerations (e.g., running the system with a single Control script) also led to simplified user-facing logic in the Control Module and Output Module (simply running the command `./run.sh`).

5 Economic Considerations (LO23)

2D Localizer is not currently intended for commercial sale. Its primary audience is academic or industrial researchers, particularly those working with range and vision sensors. As such, the program acts more as an open-source utility than a market-facing product.

In its current state, the software could be released under a permissive open-source license or an academic license that permits reuse for research purposes. In the future, the project may adopt a restricted license model, where interested users must obtain permission for use, especially if the system evolves into a full toolkit with specialized localization modules.

There would likely be minimal direct marketing involved, as the platform would rely on academic citations, GitHub visibility, and word-of-mouth within robotics and localization research communities. There is also the possibility to publish a research paper that uses it to showcase its performance. The cost to distribute and maintain the tool is low due to its Python-based modularity and limited dependency footprint (e.g., GTSAM, matplotlib). Infrastructure such as GitHub Actions already automates code linting and testing.

Potential users would include graduate students, robotics researchers, and developers working in simulation environments. Since localization research is a core topic in robotics, the potential user base extends to hundreds of labs and researchers globally. While monetization is not currently a focus, the value of the project lies in its reusability, transparency, and adaptability to a variety of sensor configurations and test environments.

6 Reflection on Project Management (LO24)

6.1 How Does Your Project Management Compare to Your Development Plan

Project management for 2D Localizer was largely guided by GitHub. GitHub Issues were used to track feedback and plan changes across the SRS, design documents, and VnV Plan. These issues helped break down large tasks into manageable checkpoints and provided a clear view of what needed to be addressed. The VnV Plan itself also acted as a planning tool, especially for organizing test coverage, test types, and scope.

While the technology used stayed consistent with the initial plan (e.g., Python, GTSAM, PyTest), the actual process of test and coverage planning evolved significantly during development. Adjustments were made along the way as new insights were gained, particularly in handling multiple types of input files.

6.2 What Went Well?

One of the highlights of the project was setting up a consistent development environment. I used a Makefile to streamline environment setup and enforce dependency management. I also integrated flake8 to enforce PEP8 coding guidelines, which helped keep the codebase clean and easier to maintain. The GitHub workflow with Issues and commits also kept things traceable and organized.

6.3 What Went Wrong?

Unit testing took longer than expected to finalize which meant prioritizing modules to test. Since many parts of the system relied on different input files (e.g., sensor data, simulation outputs), it was tricky to design tests that covered all meaningful scenarios on top of verifying their data type. The need to support a variety of file-based inputs made test coverage more involved than anticipated.

6.4 What Would you Do Differently Next Time?

If I were to do this project again, I would aim to finalize more of the core codebase earlier. This would allow for a more deliberate and gradual setup of unit tests rather than rushing to implement and validate in parallel. Getting a stable baseline sooner would also make it easier to catch edge cases early and build stronger test coverage over time.