Module Interface Specification for 2D Localizer

Aliyah Jimoh

March 17, 2025

1 Revision History

Date	Version	Notes
2025/03/19	1.0	Initial Draft

2 Symbols, Abbreviations and Acronyms

See SRS Documentation at https://github.com/AliyahJimoh/2D-Localizer/blob/main/docs/SRS/SRS.pdf

[Also add any additional symbols, abbreviations or acronyms —SS]

Contents

3 Introduction

The following document details the Module Interface Specifications for 2D Localizer, a program that implements various sensors to help localize mobile robots on a 2D plane in enclosed environments.

Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at https://github.com/AliyahJimoh/2D-Localizer.

4 Notation

[You should describe your notation. You can use what is below as a starting point. —SS]

The structure of the MIS for modules comes from Hoffman and Strooper (1995), with the addition that template modules have been adapted from Ghezzi et al. (2003). The mathematical notation comes from Chapter 3 of Hoffman and Strooper (1995). For instance, the symbol := is used for a multiple assignment statement and conditional rules follow the form $(c_1 \Rightarrow r_1|c_2 \Rightarrow r_2|...|c_n \Rightarrow r_n)$.

The following table summarizes the primitive data types used by 2D Localizer.

Data Type	Notation	Description
character	char	a single symbol or digit
integer	$\mathbb{Z}$	a number without a fractional component in (-∞, ∞)
natural number	$\mathbb{N}$	a number without a fractional component in [1, ∞)
real	$\mathbb{R}$	any number in (-∞, ∞)

The specification of 2D Localizer uses some derived data types: sequences, strings, and tuples. Sequences are lists filled with elements of the same data type. Strings are sequences of characters. Tuples contain a list of values, potentially of different types. In addition, 2D Localizer uses functions, which are defined by the data types of their inputs and outputs. Local functions are described by giving their type signature followed by their specification.

5 Module Decomposition

The following table is taken directly from the Module Guide document for this project.

Level 1	**Level 2**
Hardware-Hiding Module	
Behaviour-Hiding Module	Input Format Module Output Format Module Localization Module Control Module Accuracy Evaluation Module
Software Decision Module	Plotting Module

Table 1: Module Hierarchy

6 MIS of Control Module

6.1 Module

main

6.2 Uses

- Input Format Module (Section 7)
- Localization Module (Section 8)
- Plotting Module (Section 11)

6.2.1 Exported Constants

None

6.2.2 Exported Access Programs

Name	In	Out	Exceptions
main	-	-	Runtime error

6.3 Semantics

6.3.1 State Variables

None

6.3.2 Environment Variables

- user_input: stores user-provided input

6.3.3 Assumptions

6.3.4 Access Routine Semantics

main():

- transition: None
- output: The control module is the first to be called
- exception: None

Get(user_input) variables from user

load_input(user_input)

 # verify_input(All inputs from file)

localize(beacons, T_{mf}, T_{rf}, range_measurements)

 # plot(estimated poses, beacons, T_{mf})

7 MIS of Input Format Module

7.1 Module

input_format

7.2 Uses

None

7.3 Syntax

7.3.1 Exported Constants

None

7.3.2 Exported Access Programs

Name	In	Out	Exceptions
load_input	String	*Data*	File error (file does not exist)
verify_input	-	-	Error in ___variable

7.4 Semantics

7.4.1 State Variables

- sensor_data(user_input):
  - range_measurements: $\mathbb{R}^N$
  - camera: $\mathbb{R}^3$

7.4.2 Environment Variables

None

7.4.3 Assumptions

- The module will call on a yaml file

7.4.4 Access Routine Semantics

load_input("Filename" + ".yaml"):

- output:

  - beacons: $\mathbb{R}^{N\times 2}$
  - fiducial_map: $\mathbb{R}^3$
  - fiducial_robot: $\mathbb{R}^3$
  - range_measurements: $\mathbb{R}^N$
- exception: *File not found*

verify_input(data):

- exception: *Data format was incorrect*

8 MIS of Localization Module

8.1 Module

localization

8.2 Uses

- Input Format Module (Section 7)

8.3 Syntax

8.3.1 Exported Constants

None

8.3.2 Exported Access Programs

Name	In	Out	Exceptions
localize	User Data	$\mathbb{R}^3$	-

8.4 Semantics

8.4.1 State Variables

initial/current pose

8.4.2 Environment Variables

None

8.4.3 Assumptions

- GTSAM is installed

8.4.4 Access Routine Semantics

localize(beacons, fm_map, fm_robot, range_m):

- output: Estimated pose of the robot
  - estimated_pose: $\mathbb{R}^3$
- exception: *Format errors*

8.4.5 Local Functions

GTSAM functions

9 MIS of Accuracy Evaluation Module

9.1 Module

accuracy

9.2 Uses

- Localization Module (Section 8)

9.3 Syntax

9.3.1 Exported Constants

None

9.3.2 Exported Access Programs

Name	In	Out	Exceptions
compute_fim	$(\mathbb{R}^2, \mathbb{R}^{N\times 2}, \mathbb{R}^N)$	$\mathbb{R}^{2\times 2}$	-
compute_crlb	$\mathbb{R}^{2\times 2}$	$\mathbb{R}^{2\times 2}$	-

9.4 Semantics

9.4.1 State Variables

None

9.4.2 Environment Variables

None

9.4.3 Assumptions

- Noise variances are positive

9.4.4 Access Routine Semantics

compute_fim(estimated_pose, beacons, range_variances):

- output: A 2×2 Fisher Information Matrix (FIM), computed as:

$$\mathcal{I}(\hat{\mathbf{x}}) = \sum_{j=1}^{N} \frac{1}{\sigma_j^2} \frac{(\hat{\mathbf{x}} - \mathbf{a}_j)(\hat{\mathbf{x}} - \mathbf{a}_j)^T}{\|\hat{\mathbf{x}} - \mathbf{a}_j\|^2}$$

compute_fim(estimated_pose, beacons, range_variances):

- output: A 2×2 CRLB matrix, computed as:

$$\mathcal{C} = \mathcal{I}^{-1}$$

- exception: *If $\mathcal{I}$ is singular, the function returns 'None'.*

9.4.5 Local Functions

None

10 MIS of Output Module

10.1 Module

output

10.2 Uses

- Localization Module (Section 8)

10.3 Syntax

10.3.1 Exported Constants

None

10.3.2 Exported Access Programs

Name	In	Out	Exceptions
output_format	-	-	-
output_pose	-	-	-

10.4 Semantics

10.4.1 State Variables

None

10.4.2 Environment Variables

None

10.4.3 Assumptions

-

10.4.4 Access Routine Semantics

output_format():

- transition: [if appropriate —SS]
- output: [if appropriate —SS]
- exception: [if appropriate —SS]

output_pose():

- transition: [if appropriate —SS]
- output: [if appropriate —SS]
- exception: [if appropriate —SS]

10.4.5 Local Functions

None

11 MIS of Plotting Module

11.1 Module

plot

11.2 Uses

11.3 Syntax

11.3.1 Exported Constants

11.3.2 Exported Access Programs

Name	In	Out	Exceptions
plot_localization_live	$R^{N\times 2}, R^2$, Image	Plot	-
update_trajectory	R^2	-	-

11.4 Semantics

11.4.1 State Variables

None

11.4.2 Environment Variables

None

11.4.3 Assumptions

None

11.4.4 Access Routine Semantics

plot_localization_live(beacons, fm_map_2D, map):

- output: A dynamic plot showing real-time robot localization.

update_trajectory(estimated_pose):

- transition: Changes the robot's position on the map

11.4.5 Local Functions

None

References

Carlo Ghezzi, Mehdi Jazayeri, and Dino Mandrioli. *Fundamentals of Software Engineering*. Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, 2003.

Daniel M. Hoffman and Paul A. Strooper. *Software Design, Automated Testing, and Maintenance: A Practical Approach*. International Thomson Computer Press, New York, NY, USA, 1995. URL http://citeseer.ist.psu.edu/428727.html.

12 Appendix

[Extra information if required —SS]