Software Requirements Specification for 2D Localizer

Aliyah Jimoh

February 5, 2025

Contents

1 Reference Material **iv**
- 1.1 Table of Units . . . iv
- 1.2 Table of Symbols . . . iv
- 1.3 Abbreviations and Acronyms . . . v
- 1.4 Mathematical Notation . . . v

2 Introduction **1**
- 2.1 Purpose of Document . . . 1
- 2.2 Scope of Requirements . . . 1
- 2.3 Characteristics of Intended Reader . . . 1
- 2.4 Organization of Document . . . 2

3 General System Description **2**
- 3.1 System Context . . . 2
- 3.2 User Characteristics . . . 3
- 3.3 System Constraints . . . 3

4 Specific System Description **3**
- 4.1 Problem Description . . . 3
  - 4.1.1 Terminology and Definitions . . . 3
  - 4.1.2 Physical System Description . . . 3
  - 4.1.3 Goal Statements . . . 4
- 4.2 Solution Characteristics Specification . . . 4
  - 4.2.1 Assumptions . . . 4
  - 4.2.2 Theoretical Models . . . 4
  - 4.2.3 General Definitions . . . 8
  - 4.2.4 Data Definitions . . . 9
  - 4.2.5 Data Types . . . 9
  - 4.2.6 Instance Models . . . 10
  - 4.2.7 Input Data Constraints . . . 12
  - 4.2.8 Properties of a Correct Solution . . . 13

5 Requirements **13**
- 5.1 Functional Requirements . . . 14
- 5.2 Nonfunctional Requirements . . . 14
- 5.3 Rationale . . . 14

6 Likely Changes **15**

7 Unlikely Changes **15**

8 Traceability Matrices and Graphs **15**

9 Development Plan **18**

10 Values of Auxiliary Constants **18**

Revision History

Date	Version	Notes
2025/02/05	1.0	Initial Draft
Date 2	1.1	Notes

1 Reference Material

This section records information for easy reference.

1.1 Table of Units

Throughout this document SI (Système International d'Unités) is employed as the unit system. In addition to the basic units, several derived units are used as described below. For each unit, the symbol is given followed by a description of the unit and the SI name.

symbol	**unit**	**SI**
m	length	metre
s	time	second
°	angle	degrees

1.2 Table of Symbols

The table that follows summarizes the symbols used in this document along with their units. The symbols are listed in alphabetical order.

[Remeber to organize them in order —TPLT]

symbol	**unit**	**description**
i	–	Index for robot's poses
j	–	Index for number of beacons
η_j	–	Sensor noise
n	–	Index for number of FMs
S	–	Set of beacon coordinates
$g_j(x_i)$	m	Noisy range measurement of the robot's ith taken from jth beacon
A	m^2	Area of the 2D environment
F	–	Coordinates of FMs
σ_j^2	m^2	Noise variance for jth beacon
$\tilde{d}_j$	m	Measured distance with noise
$C(x_i)$	m	Cost Function
I	m^-2	Total Fisher Information Matrix
θ	deg	Robot angle
T_{robot}	m^2	surface area over which heat is transferred in
$\tilde{D}$	m^2	coil surface area

d_j	m^2	surface area over which heat is transferred in
T_{env}	m^2	coil surface area
A_{in}	m^2	surface area over which heat is transferred in

1.3 Abbreviations and Acronyms

Abbreviation/Acronym	Definition
2D	Two-Dimensional
A	Assumption
CRLB	Cramér-Rao Lower Bound
DD	Data Definition
FIM	Fisher Information Matrix
FM	Fiducial Marker
GD	General Definition
GS	Goal Statement
IM	Instance Model
LC	Likely Change
Localizer	2D Localization Solution
PS	Physical System Description
R	Requirement
SE(2)	Special Euclidean Group in 2D
SRS	Software Requirements Specification
TM	Theoretical Model

1.4 Mathematical Notation

Throughout this document, there will be typographic conventions as well as mathematical operators that are used to distinguish different variables and operations.

Variable	Definition	Description
$\mathbf{A}$	Matrix	Bold capital letter
$\mathbf{a}$	Vector	Bold lowercase letter
a	Scalar	Lowercase letter

2 Introduction

Mobile robots have been used to traverse areas with various hazards, help collect data whether through its sensors or obtaining samples, and overall complete challenging tasks. Due to their autonomy, there is no need to constantly monitor them as they have their own means with interacting with the environment. However, it could raise some concern when there is no reliable method to track their movements, especially if they are placed in a vastly large area or places that may be difficult to access once they are operational. Risks such as having difficulty retrieving them if there is a malfunction keeping them from returning or possible collisions in the area or with other robots are reasons one would want to find a way to locate them as they carry out their tasks. The program being documented, 2D Localizer, proposes to solve this problem by developing a 2D localization solution that can implement various sensors to accurately localize the robots as they traverse the map provided.

The following section provides an overview of the Software Requirement Specification (SRS) for 2D Localizer. This section explains the purpose of this document, the scope of the requirements, the characteristics of the intended reader, and the organization of the document.

2.1 Purpose of Document

The main purpose of this document is to describe the requirements needed to run the localizer. Information such as the constraints, assumptions and theoretical models used will be provided to help readers get a better understanding of the purpose and computations of 2D Localizer. Therefore, it can be used as a reference guide on how to plan and set up the requirements needed by the user to get the desired and accurate results.

2.2 Scope of Requirements

The scope of the requirements includes the robot analyzed being in a controlled environment to help with potential lighting problems for vision sensors used. The sensors used on the robot and environment are specified for modelling purposes. Some models can include a sum through sensors having independent measurements. The noise from the sensors will be considered zero-mean Gaussian for simplicity in calculations.

2.3 Characteristics of Intended Reader

Reviewers of this documentation should have taken a graduate course on linear algebra, estimation theory or matrix computation. They should also have some knowledge in undergraduate statistics and probability.

2.4 Organization of Document

The organization of this document follows the template for an SRS for scientific computing software proposed by Smith and Lai (2005), Smith et al. (2007), and Smith and Koothoor (2016). The presentation follows the standard pattern of presenting goals, theories, definitions, and assumptions. For readers that would like a more bottom up approach, they can start reading the data definitions and trace back to find any additional information they require.

The goal statements are refined to the theoretical models and the theoretical models to the instance models. The data definitions are used to support the definitions of the different models.

3 General System Description

This section provides general information about the system. It identifies the interfaces between the system and its environment, describes the user characteristics and lists the system constraints.

3.1 System Context

The system context is displayed in Figure 1 below. The circles represent the external aspects related to the software which are the users. The rectangle represents the software system being used (2D Localizer) and the arrows explain what information is being passed between the user and the software.

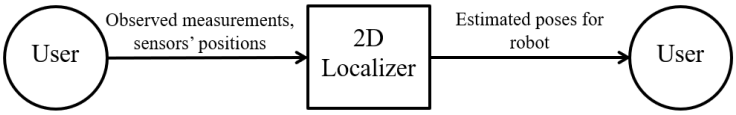

Figure 1: System Context

- User Responsibilities:
  - Provide inputs including the coordinates of each environmental sensor and the measurements taken from them.
  - Evaluate the inputs to ensure all are their respective types
- ProgName Responsibilities:
  - Detect data type mismatch, such as a string of characters instead of a floating point number
  - Estimate the position and orientation of the robot

3.2 User Characteristics

The end user of 2D Localizer should have some familiarity with types of range sensors along with how to read and collect their data. They should also have basic experience with programming.

3.3 System Constraints

There are no system constraints.

4 Specific System Description

This section first presents the problem description, which gives a high-level view of the problem to be solved. This is followed by the solution characteristics specification, which presents the assumptions, theories, definitions and finally the instance models.

4.1 Problem Description

2D Localizer is intended to help keep track of mobile robots while carrying out their tasks.

4.1.1 Terminology and Definitions

This subsection provides a list of terms that are used in the subsequent sections and their meaning, with the purpose of reducing ambiguity and making it easier to correctly understand the requirements:

- **Pose:** Position and orientation of the robot.
- **Localization**: Determining where an object is with respect to its environment.
- **Fiducial Markers**: Markers placed around the environment for the robot to determine its location.
- **Beacon**: A range sensor

4.1.2 Physical System Description

The physical system of the localizer includes the following elements:

PS1: The mobile robot

PS2: The beacons placed in the environment

PS3: The camera sensors on the mobile robot

PS4: The fiducial markers in the environment

4.1.3 Goal Statements

Given the imported 2D map, coordinates of all sensors and fiducial markers in the environment, and the noisy measurements of the sensors, the goal statements are:

GS1: Calculate the estimated pose and error of the robot throughout its trajectory from both sensors (environment and robot).

GS2: Display a visual representation of the robot traversing the 2-D map with the sensors placed.

4.2 Solution Characteristics Specification

The instance models that govern 2D Localizer are presented in Subsection 4.2.6. The information to understand the meaning of the instance models and their derivation is also presented, so that the instance models can be verified.

4.2.1 Assumptions

This section simplifies the original problem and helps in developing the theoretical model by filling in the missing information for the physical system. The numbers given in the square brackets refer to the theoretical model [TM], general definition [GD], data definition [DD], instance model [IM], or likely change [LC], in which the respective assumption is used.

A1 : Robot uses camera sensors while the environment uses beacons and fiducial markers

A2 : Localizer is used in a controlled environment (i.e., indoors)

A1 : Each sensor has independent measurements

A1 : Sensor noise is zero-mean Gaussian

4.2.2 Theoretical Models

This section focuses on the general equations and laws that 2D Localizer is based on.

Number	TM1
Label	**Noisy Range Measurement**
Equation	$$g_j(x) = \|\mathbf{x_i} - \mathbf{a_j}\| + \eta_j$$
Description	The equation above gives the noisy range measurement g_j (m) of the beacons placed in the environment (A1) where $\mathbf{x_i}$ is the position of the robot (m), $\mathbf{a_j}$ is the position of the jth beacon placed (m), and η_j is the noise from the jth beacon (m) (A4).
Source	Sequeira et al. (2024)
Ref. By	DD1
Preconditions	A4
Derivation	None

Number	TM2
Label	**Cost Function**
Equation	$$C(x) = \sum_{j=1}^{N} \frac{1}{\sigma_j^2} \left(\tilde{d}_j - \|\mathbf{x_i} - \mathbf{a_j}\| \right)^2$$
Description	The above equation gives the error between the measured and predicted ranges, also known as the cost function. $\tilde{d}_j$ is the real measurement with noise from the sensors (DD1) σ_j^2 is the sensor variance used in the noise factor $\|\mathbf{x_i} - \mathbf{a_j}\|$ is the predicted range the robot is from the jth beacon
Source	-
Ref. By	None
Preconditions	A4
Derivation	None

Number	TM3
Label	**Fisher Information Matrix**
Equation	$$\mathbf{I} \cong \sum_{j=1}^{N} \frac{1}{\sigma_j^2} \frac{(\widehat{x} - a_j)(\widehat{x} - a_j)^T}{\|\widehat{x} - a_j\|^2}$$
Description	This equation gives the Fisher Information Matrix which shows how much information the data provides about the robot's position σ_j^2 is the sensor variance used in the noise factor $\widehat{x}$ is the estimated position in IM2
Source	Citation here
Ref. By	TM4
Preconditions	A4, IM2
Derivation	

Number	TM4
Label	**Cramér-Rao Lower Bound**
Equation	$$Var(\hat{\theta}) \geq \frac{1}{I(\theta)}$$
Description	This equation gives the MLE which estimates the best position $\widehat{x}$ (m) to align based on the range measurements given.
Source	Citation here
Ref. By	None
Preconditions	TM3
Derivation	

Number	TM5
Label	**SE(2) Transformation**
Equation	$$T_{robot} = \begin{bmatrix} R(\theta) & t \\ 0 & 1 \end{bmatrix}$$
Description	This equation gives the MLE which estimates the best position $\widehat{x}$ (m) to align based on the range measurements given.
Source	Citation here
Ref. By	DD**??**, DD**??**
Preconditions	
Derivation	

4.2.3 General Definitions

[General Definitions (GDs) are a refinement of one or more TMs, and/or of other GDs. The GDs are less abstract than the TMs. Generally the reduction in abstraction is possible through invoking (using/referencing) Assumptions. For instance, the TM could be Newton's Law of Cooling stated abstracting. The GD could take the general law and apply it to get a 1D equation. —TPLT]

This section collects the laws and equations that will be used in building the instance models.

[Some projects may not have any content for this section, but the section heading should be kept. —TPLT] [Modify the examples below for your problem, and add additional definitions as appropriate. —TPLT]

Number	GD1
Label	**Probability Density Function**
SI Units	m^{-1}
Equation	$$p_j\left(\tilde{d}_j \mid x\right) = \mathcal{N}\left(\|x - a_j\|, \sigma_j^2\right)$$
Description	Newton's law of cooling describes convective cooling from a surface. The law is stated as: the rate of heat loss from a body is proportional to the difference in temperatures between the body and its surroundings. $q(t)$ is the thermal flux ($\mathrm{W\,m^{-2}}$). h is the heat transfer coefficient, assumed independent of T (A**??**) ($\mathrm{W\,m^{-2}\,{}^{\circ}C^{-1}}$). $\Delta T(t) = T(t) - T_{\mathrm{env}}(t)$ is the time-dependent thermal gradient between the environment and the object (°C).
Source	Citation here
Ref. By	DD**??**, DD**??**

Detailed derivation of simplified rate of change of temperature

[This may be necessary when the necessary information does not fit in the description field. —TPLT] [Derivations are important for justifying a given GD. You want it to be clear where the equation came from. —TPLT]

4.2.4 Data Definitions

[The Data Definitions are definitions of symbols and equations that are given for the problem. They are not derived; they are simply used by other models. For instance, if a problem depends on density, there may be a data definition for the equation defining density. The DDs are given information that you can use in your other modules. —TPLT]

[All Data Definitions should be used (referenced) by at least one other model. —TPLT]

This section collects and defines all the data needed to build the instance models. The dimension of each quantity is also given. [Modify the examples below for your problem, and add additional definitions as appropriate. —TPLT]

Number	DD1
Label	**Actual Range Measurement**
Symbol	$\tilde{d}_j$
SI Units	m
Equation	$\tilde{d}_j = \|\mathbf{x_i} - \mathbf{a_j}\| + \eta_j$
Description	T_C is the temperature of the coil (°C). T_W is the temperature of the water (°C). The heat flux out of the coil, q_C ($\mathrm{W\,m^{-2}}$), is found by assuming that Newton's Law of Cooling applies (A**??**). This law (GD**??**) is used on the surface of the coil, which has area A_C ($\mathrm{m^2}$) and heat transfer coefficient h_C ($\mathrm{W\,m^{-2}\,°C^{-1}}$). This equation assumes that the temperature of the coil is constant over time (A**??**) and that it does not vary along the length of the coil (A**??**).
Sources	Citation here
Ref. By	TM**??**

4.2.5 Data Types

[This section is optional. In many scientific computing programs it isn't necessary, since the inputs and outpus are straightforward types, like reals, integers, and sequences of reals and integers. However, for some problems it is very helpful to capture the type information. —TPLT]

[The data types are not derived; they are simply stated and used by other models. —TPLT]

[All data types must be used by at least one of the models. —TPLT]

[For the mathematical notation for expressing types, the recommendation is to use the notation of **?**. —TPLT]

This section collects and defines all the data types needed to document the models. [Modify the examples below for your problem, and add additional definitions as appropriate. —TPLT]

Type Name	Name for Type
Type Def	mathematical definition of the type
Description	description here
Sources	Citation here, if the type is borrowed from another source

4.2.6 Instance Models

[The motivation for this section is to reduce the problem defined in "Physical System Description" (Section 4.1.2) to one expressed in mathematical terms. The IMs are built by refining the TMs and/or GDs. This section should remain abstract. The SRS should specify the requirements without considering the implementation. —TPLT]

This section transforms the problem defined in Section 4.1 into one which is expressed in mathematical terms. It uses concrete symbols defined in Section 4.2.4 to replace the abstract symbols in the models identified in Sections 4.2.2 and 4.2.3.

The goals [reference your goals —TPLT] are solved by [reference your instance models —TPLT]. [other details, with cross-references where appropriate. —TPLT] [Modify the examples below for your problem, and add additional models as appropriate. —TPLT]

Number	IM1
Label	**Full Likelihood Function**
Input	$\tilde{D}$, N
Output	$p(\tilde{D}\|x)$
Equation	$$p\left(\tilde{D}\|x\right) = \prod_{j=1}^{N} p\left(\tilde{d}_j\|x\right)$$
Description	T_W is the water temperature (°C). T_P is the PCM temperature (°C). T_C is the coil temperature (°C). $\tau_W = \frac{m_W C_W}{h_C A_C}$ is a constant (s). $\eta = \frac{h_P A_P}{h_C A_C}$ is a constant (dimensionless). The above equation applies as long as the water is in liquid form, $0 < T_W < 100^o$C, where 0^oC and 100^oC are the melting and boiling points of water, respectively (A**??**, A**??**).
Sources	Citation here
Ref. By	IM**??**

Number	IM2
Label	**Maximum Likelihood Estimation**
Input	$p(\tilde{D}\|x)$
Output	$\widehat{x}$
Equation	$\widehat{x} = \arg\min p(\tilde{D}\|x)$
Description	T_W is the water temperature (°C). T_P is the PCM temperature (°C). T_C is the coil temperature (°C). $\tau_W = \frac{m_W C_W}{h_C A_C}$ is a constant (s). $\eta = \frac{h_P A_P}{h_C A_C}$ is a constant (dimensionless). This equation gives the MLE which estimates the best position $\widehat{x}$ (m) to align based on the range measurements given.
Sources	Citation here
Ref. By	IM??

Derivation of ...

[The derivation shows how the IM is derived from the TMs/GDs. In cases where the derivation cannot be described under the Description field, it will be necessary to include this subsection. —TPLT]

4.2.7 Input Data Constraints

Table 5 shows the data constraints on the input output variables. The column for physical constraints gives the physical limitations on the range of values that can be taken by the variable. The column for software constraints restricts the range of inputs to reasonable values. The software constraints will be helpful in the design stage for picking suitable algorithms. The constraints are conservative, to give the user of the model the flexibility to experiment with unusual situations. The column of typical values is intended to provide a feel for a common scenario. The uncertainty column provides an estimate of the confidence with which the physical quantities can be measured. This information would be part of the input if one were performing an uncertainty quantification exercise.

The specification parameters in Table 5 are listed in Table 7.

Table 5: Input Variables

Var	Physical Constraints	Software Constraints	Typical Value	Uncertainty
L	$L > 0$	$L_{\text{min}} \leq L \leq L_{\text{max}}$	1.5 m	10%

(*) [you might need to add some notes or clarifications —TPLT]

Table 7: Specification Parameter Values

Var	Value
L_{min}	0.1 m

4.2.8 Properties of a Correct Solution

A correct solution must exhibit [fill in the details —TPLT]. [These properties are in addition to the stated requirements. There is no need to repeat the requirements here. These additional properties may not exist for every problem. Examples include conservation laws (like conservation of energy or mass) and known constraints on outputs, which are usually summarized in tabular form. A sample table is shown in Table 9 —TPLT]

Table 9: Output Variables

Var	Physical Constraints
T_W	$T_{\text{init}} \leq T_W \leq T_C$ (by A**??**)

[This section is not for test cases or techniques for verification and validation. Those topics will be addressed in the Verification and Validation plan. —TPLT]

5 Requirements

This section provides the functional requirements, the business tasks that the software is expected to complete, and the nonfunctional requirements, the qualities that the software is expected to exhibit.

5.1 Functional Requirements

R1: Provide the inputs for 2D Localizer which include the 2D map, the coordinates of all sensors and markers located in the environment, and the measurements taken from the sensors.

R2: 2D Localizer will acquire $\widehat{x}$ from the inputs in R1.

R3: 2D Localizer will calculate the estimated posed from the measurements provided for IM2.

R4: 2D Localizer will verify that inputs provided are within their constraints.

R5: 2D Localizer will show a visual animated graph that tracks the mobile robots work while also displaying the sensors' coordinates.

5.2 Nonfunctional Requirements

This problem priorities the accuracy of the estimations along with its performance when it comes to keeping up with the robot's movements. That being said, the nonfunctional requirements are:

NFR1: **Accuracy**: The accuracy of the computed estimations should meet the threshold when comparing to the predicted measurements.

NFR2: **Understandability**: The software should be simple to understand and implement.

NFR3: **Maintainability**: The software should be simple modify some components when needed.

NFR4: **Usability**: The software should be easy to run and have minimal troubleshooting.

5.3 Rationale

This program has rationale for the assumptions mentioned in 4.2.1:

- A1: Assuming the type of sensors assisted in setting up the type of output variables that is needed for each set of estimates.
- A2: Referring to A1, vision sensors will be used on the robot and lighting could affect the way they detect FMs. Another main reason for this assumption is so that the coordinates of each beacon and FM can be structured around the area the user provides.
- A3: In order to find the sum and products of variables that need data from each beacon and FM places (TM3, IM1), having each sensor collect independent data from different positions would help with the overall accuracy the program want to achieve.

- A4: Having a Gaussian noise assumption simplifies computations in the program especially when some sensors used in robotics follow a Gaussian distribution.

6 Likely Changes

LC1: [Give the likely changes, with a reference to the related assumption (aref), as appropriate. —TPLT]

7 Unlikely Changes

LC2: [Give the unlikely changes. The design can assume that the changes listed will not occur. —TPLT]

8 Traceability Matrices and Graphs

The purpose of the traceability matrices is to provide easy references on what has to be additionally modified if a certain component is changed. Every time a component is changed, the items in the column of that component that are marked with an "X" may have to be modified as well. Table 11 shows the dependencies of theoretical models, general definitions, data definitions, and instance models with each other. Table 12 shows the dependencies of instance models, requirements, and data constraints on each other. Table 13 shows the dependencies of theoretical models, general definitions, data definitions, instance models, and likely changes on the assumptions.

[You will have to modify these tables for your problem. —TPLT]

[The traceability matrix is not generally symmetric. If GD1 uses A1, that means that GD1's derivation or presentation requires invocation of A1. A1 does not use GD1. A1 is "used by" GD1. —TPLT]

[The traceability matrix is challenging to maintain manually. Please do your best. In the future tools (like Drasil) will make this much easier. —TPLT]

The purpose of the traceability graphs is also to provide easy references on what has to be additionally modified if a certain component is changed. The arrows in the graphs represent dependencies. The component at the tail of an arrow is depended on by the component at the head of that arrow. Therefore, if a component is changed, the components that it points to should also be changed. Figure **??** shows the dependencies of theoretical models, general definitions, data definitions, instance models, likely changes, and assumptions on each other. Figure **??** shows the dependencies of instance models, requirements, and data constraints on each other.

	TM??	TM??	TM??	GD??	GD??	DD??	DD??	DD??	DD??	IM??	IM??	IM?
TM??												
TM??			X									
TM??												
GD??												
GD??	X											
DD??				X								
DD??				X								
DD??												
DD??								X				
IM??					X	X	X				X	
IM??					X		X		X	X		
IM??		X										
IM??		X	X				X	X	X		X	

Table 11: Traceability Matrix Showing the Connections Between Items of Different Sections

	IM??	IM??	IM??	IM??	4.2.7	R??	R??
IM??		X				X	X
IM??	X			X		X	X
IM??						X	X
IM??		X				X	X
R??							
R??						X	
R??					X		
R2	X	X				X	X
R??	X						
R??		X					
R??			X				
R??				X			
R4			X	X			
R??		X					
R??		X					

Table 12: Traceability Matrix Showing the Connections Between Requirements and Instance Models

	A1	A2	A3	A4
TM1	X			
TM2				
TM3				
TM4				
TM5				
GD1		X		
DD1				
IM1				
IM2				
LC??				
LC??				
LC??				
LC??				
LC??				
LC??				

Table 13: Traceability Matrix Showing the Connections Between Assumptions and Other Items

9 Development Plan

[This section is optional. It is used to explain the plan for developing the software. In particular, this section gives a list of the order in which the requirements will be implemented. In the context of a course this is where you can indicate which requirements will be implemented as part of the course, and which will be "faked" as future work. This section can be organized as a prioritized list of requirements, or it could should the requirements that will be implemented for "phase 1", "phase 2", etc. —TPLT]

10 Values of Auxiliary Constants

[Show the values of the symbolic parameters introduced in the report. —TPLT]

[The definition of the requirements will likely call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance. —TPLT]

[The value of FRACTION, for the Maintainability NFR would be given here. —TPLT]

References

Ethan Sequeira, Hussein Saad, Stephen Kelly, and Matthew Giamou. Towards Optimal Beacon Placement for Range-Aided Localization. *Proceedings of the Conference on Robots and Vision*, may 28 2024. https://crv.pubpub.org/pub/jl15869x.

W. Spencer Smith and Nirmitha Koothoor. A document-driven method for certifying scientific computing software for use in nuclear safety analysis. *Nuclear Engineering and Technology*, 48(2):404–418, April 2016. ISSN 1738-5733. doi: http://dx.doi.org/10.1016/j.net.2015.11.008. URL http://www.sciencedirect.com/science/article/pii/S1738573315002582.

W. Spencer Smith and Lei Lai. A new requirements template for scientific computing. In J. Ralyté, P. Ågerfalk, and N. Kraiem, editors, *Proceedings of the First International Workshop on Situational Requirements Engineering Processes – Methods, Techniques and Tools to Support Situation-Specific Requirements Engineering Processes, SREP'05*, pages 107–121, Paris, France, 2005. In conjunction with 13th IEEE International Requirements Engineering Conference.

W. Spencer Smith, Lei Lai, and Ridha Khedri. Requirements analysis for engineering computation: A systematic approach for improving software reliability. *Reliable Computing, Special Issue on Reliable Engineering Computation*, 13(1):83–107, February 2007.

[The following is not part of the template, just some things to consider when filing in the template. —TPLT]
[Grammar, flow and LaTeXadvice:

- For Mac users `*.DS_Store` should be in `.gitignore`
- LaTeX and formatting rules
  - Variables are italic, everything else not, includes subscripts (link to document)
    - Conventions
    - Watch out for implied multiplication
  - Use BibTeX
  - Use cross-referencing
- Grammar and writing rules
  - Acronyms expanded on first usage (not just in table of acronyms)
  - "In order to" should be "to"

—TPLT]
[Advice on using the template:

- Difference between physical and software constraints
- Properties of a correct solution means *additional* properties, not a restating of the requirements (may be "not applicable" for your problem). If you have a table of output constraints, then these are properties of a correct solution.
- Assumptions have to be invoked somewhere
- "Referenced by" implies that there is an explicit reference
- Think of traceability matrix, list of assumption invocations and list of reference by fields as automatically generatable
- If you say the format of the output (plot, table etc), then your requirement could be more abstract

—TPLT]