System Verification and Validation Plan for IP Simulator

Mina Mahdipour

March 8, 2023

1 Revision History

Date	Version	Notes
February 13, 2023	1.0	The first version of VnV
February 13, 2023	1.1	Added plan part
February 17, 2023	1.2	Added system test description
February 18, 2023	1.3	Updated test section
February 19, 2023	1.4	Refined the whole document
February 20, 2023	1.5	Updated NFRs test casee

Contents

1 Revision History **i**

2 Symbols, Abbreviations and Acronyms **iv**

3 General Information **1**
3.1 Summary 1
3.2 Objectives 1
3.3 Relevant Documentation 1

4 Plan **2**
4.1 Verification and Validation Team 2
4.2 SRS Verification Plan 2
4.3 Design Verification Plan 3
4.4 Verification and Validation Plan Verification Plan 3
4.5 Implementation Verification Plan 3
4.6 Automated Testing and Verification Tools 3
4.7 Software Validation Plan 4

5 System Test Description **4**
5.1 Tests for Functional Requirements 4
5.1.1 Input validation 4
5.1.2 Output Validity 7
5.2 Tests for Nonfunctional Requirements 8
5.2.1 Portability 8
5.2.2 Usability 9
5.2.3 Accuracy 9
5.3 Traceability Between Test Cases and Requirements 10

6 Unit Test Description **10**
6.1 Unit Testing Scope 10

7 Appendix **12**
7.1 Usability Survey Questions? 12

List of Tables

1 Verification and Validation Team 2
2 Input Validation Test Cases 5
3 Output Validation Test Cases 7
4 Traceability Matrix between tests and requirements 10

2 Symbols, Abbreviations and Acronyms

symbol	description
FR	Functional Requirements
MIS	Module Interface Specification
NFR	Nonfunctional Requirements
N/A	Not Applicable
VnV	Verification and Validation
SRS	Software Requirements Specification
T	Test
TC	Test Case

Section 1 of SRS document can be referred by the reader for complete symbols used within the system.

This document outlines the roadmap of the verification and validation plan for Inverted Pendulum (IP) Simulator to help ensure (but not prove) the correctness and completeness of the program. It includes a plan for testing the functional and non-functional requirements, an overview of the system tests, and an outline of the unit tests (not yet complete as they depend on the as yet incomplete MIS).

3 General Information

3.1 Summary

This document provides the validation and verification plans for the IP Simulator Software. This software aims to simulate and describe the behavior of a cart-pendulum system in the presence of different values of external force. To do that, it will calculate the cart position and pendulum's angle after applying a time varying force.

3.2 Objectives

The objectives of this document are listed below:

- Establish confidence in software reliability.
- Demonstrate adequate usability.
- Verify and validate the final product.

Performance testing, stress testing, and maintainability testing are outside of the scope of this document.

3.3 Relevant Documentation

Problem Statement presents an overview of the problem. The requirements and an outline of the solution for the IP Simulator are captured in the Software Requirements Specification. The software design information will be captured in the Module Guide. The final report of running the test which described in this document, will be shown in VnV Report.

4 Plan

This section provides a roadmap of plans for different steps of producing the final product.

4.1 Verification and Validation Team

Our team details are described in table 1:

Table 1: Verification and Validation Team

Role	**Responsibility**	**Assignee**	**Document**
Supervisor	Review and provide feedbacks	Dr. Spencer Smith	All documents
Author	Create the products and refine them by feedbacks	Mina Mahdipour	All documents
Domain Expert/ Reviewer	Review and provide feedbacks	Deesha Patel	All documents
Secondary Reviewer	Review and provide feedbacks	Maryam Valian	SRS
Secondary Reviewer	Review and provide feedbacks	Karen Wang	VnV plan
Secondary Reviewer	Review and provide feedbacks	Joachim de Fourestier	MG + MIS

4.2 SRS Verification Plan

The SRS has been independently peer-reviewed by members of the VnV team according to SRS-Checklist, consisting of the domain expert Deesha Patel and the SRS reviewer, Maryam Valian, as well as Dr. Spencer Smith. The SRS document has been published to GitHub. Any issues identified during the review were tracked and verified in Github platform.

4.3 Design Verification Plan

The design document MIS will be reviewed by the MIS review team, consisting of the domain expert, Deesha Patel and the MIS reviewer, Joachim de Fourestier, as well as Dr. Spencer Smith. The MIS document will be published to GitHub. Defects will be addressed with issues on the GitHub platform. There is also MIS-Checklist, that can be used.

4.4 Verification and Validation Plan Verification Plan

The VnV plan document will be reviewed by the VnV plan review team, consisting of the domain expert Deesha Patel and the reviewer, Karen Wang as well as Dr. Spencer Smith. This document will be published to GitHub. Defects will be addressed with issues on the GitHub platform. There is also VnV-Checklist, that can be used.

4.5 Implementation Verification Plan

The IP Simulator will be developed in Python programming language. The implementation of the software will be tested under the test cases listed in section 5. The tests will either be automated, performed manually.
Static analysis will be done using Code Walkthrough technique by the Domain expert (Deesha Patel). During a meeting, the author presents the document under review, while the domain expert will run the test cases over the code by hand. She may also ask the author any relevant questions and discuss her findings with the author. Visual Studio Code will be used for program debugging and for checking syntax errors [3].

4.6 Automated Testing and Verification Tools

Automated Testing and Verification Tools will be extensively used in the development of IP Simulator for automation at different levels. These tools include git, a distributed version-control system for tracking changes in source code during software development, Pytest framework for unit, functional, and integration automated testing, and Flake8 to check for errors, enforces coding standards, and identifies code complexity issues.

4.7 Software Validation Plan

There are no plans for the validation of the IP Simulator software.

5 System Test Description

This section will define the tests to ensure IP Simulator meets the functional requirements seen in section 5 of the SRS document for simulating inverted pendulum system. The subsections combine several requirements that are be separated based on common ideas.

5.1 Tests for Functional Requirements

This section contains the system test cases for the functional requirements which are described in the SRS.

5.1.1 Input validation

This section includes tests to verify that the software rejects invalid input values such as non-numerical values or values outside of specified range. These tests also help to make sure that the the IP Simulator accepts valid input values within specified range, FR1 and FR2 of requirements in section5 of the SRS.

Input Constraints Test

1. TC-IP-1: Valid Inputs

   Control: Automatic

   Initial State: N/A

   Input: Set of input values as 1-1 in Table 2.

   Output: x, the position of the cart and θ, the angle of the pendulum

   Test Case Derivation: Derived form IP Simulator, as a normal and valid set of inputs.

   How test will be performed: Automatic test using Pytest.

Table 2: Input Validation Test Cases

	Input					Expected Output	
TC	m_p	m_c	l	F	b	$Validity$	$ErrorMessage$
1-1	0.2	0.5	0.3	50	0.2	Yes	None
2-1	0	0.5	0.3	50	0.2	No	Error: invalid input
2-2	0.2	0	0.3	50	0.2	No	Error: invalid input
2-3	0.2	0.5	0	50	0.2	No	Error: invalid input
3-1	0.2	0.5	0.3	50	0	Yes	None
4-1	-0.2	0.5	0.3	50	0.2	No	Error: invalid input
4-2	0.2	-0.5	0.3	50	0.2	No	Error: invalid input
4-3	0.2	0.5	-0.3	50	0.2	No	Error: invalid input
4-4	0.2	0.5	0.3	-50	0.2	No	Error: invalid input
4-5	0.2	0.5	0.3	50	-0.2	No	Error: invalid input
5-1	50	0.5	0.3	50	0.2	No	Error: invalid input
5-2	0.2	50	0.3	50	0.2	No	Error: invalid input
5-3	0.2	0.5	1	50	0.2	No	Error: invalid input
5-4	0.2	0.5	0.3	200	0.2	No	Error: invalid input
6-1		0.5	0.3	50	0.2	No	Error: invalid input
6-2	0.2		0.3	50	0.2	No	Error: invalid input
6-3	0.2	0.5		50	0.2	No	Error: invalid input
6-4	0.2	0.5	0.3		0.2	No	Error: invalid input
6-5	0.2	0.5	0.3	50		No	Error: invalid input
7-1	0.2	0.5	0.3	0	0.2	Yes	None

2. TC-IP-2: Zero Inputs

   Control: Automatic

   Initial State: N/A

   Input: Set of input values as 2-1 to 2-3 in Table 2.

   Output: Error message of invalid input because of zero inputs.

   Test Case Derivation: Zero input for mass of the pendulum or mass of the cart, or length of the cart are not valid.

How test will be performed: Automatic test using Pytest.

3. TC-IP-3: Valid and Zero Inputs

   Control: Automatic

   Initial State: N/A

   Input: Set of input values as 3-1 and 7-1 in Table 2.

   Output: x, the position of the cart and θ, the angle of the pendulum

   Test Case Derivation: Derived from IP Simulator, as a normal and valid set of inputs, however it should be considered that with inputs as 7-1, the position of cart stay without change and the pendulum falls and its angle will be 180 rad.

   How test will be performed: Automatic test using Pytest.

4. TC-IP-4: Negative Inputs

   Control: Automatic

   Initial State: N/A

   Input: Set of input values as 4-1 to 4-5 in Table 2.

   Output: Error message of invalid negative input.

   Test Case Derivation: Inputs can not be negative values.

   How test will be performed: Automatic test using Pytest.

5. TC-IP-5: Out of Bound Inputs

   Control: Automatic

   Initial State: N/A

   Input: Set of input values as 5-1 through 5-4 in Table.2.

   Output: Error message of inputs because of constraints.

   Test Case Derivation: The user should prepare values for inputs in the valid range for each.

   How test will be performed: Automatic test using Pytest.

Table 3: Output Validation Test Cases

	Input					Expected Output
TC	m_p	m_c	l	F	b	*Output*
1-1	0.2	0.5	0.3	50	0.2	x the position of cart, and θ, the angle of pendulum
2-1	0	0.5	0.3	50	0.2	Error: invalid input, Put non-zero values
3-1	-0.2	0.5	0.3	50	0.2	Error: invalid input, Put non-negative values
4-1	0.2	0.5	0.3	200	0.2	Error: invalid input, Out of boundary
5-1		0.5	0.3	50	0.2	Error: invalid input, Missed input

6. TC-IP-6: Missing Inputs

   Control: Automatic

   Initial State: N/A

   Input: Set of input values as 6-1 through 6-5 in Table.2.

   Output: Error message of invalid missing input.

   Test Case Derivation: The user should provide all the inputs without missing.

   How test will be performed: Automatic test using Pytest.

5.1.2 Output Validity

This section presents tests to make confidence in error handling in the software, which means the system can handle unexpected errors gracefully and provides clear error messages when errors occur. The other goal of these tests is to verify that system simulates the cart moving back and forth and the pendulum rotates according to input values, as the R3 and R4 of requirements in section 5 of the SRS has defined.

1. TC-IP-1

   Control: Automatic

   Initial State: the software has run by the input values of 1-1 in Table 3.

   Input: input variables like 1-1 in Table 3.

Output: The x can be positive or negative, which means the cart moves left or right from the source and θ is positive or zero.

Test Case Derivation: Derived form IP Simulator, as a normal and valid set of inputs have been provided.

How test will be performed: It will be performed by IP Simulator and Pytest.

2. TC-IP-2

   Control: Automatic

   Initial State: the software has run by the input values of 2-1 through 5-1 in Table 3.

   Input: invalid inputs like 2-1 through 5-1 in Table 3.

   Output: The output is the appropriate error to clarify what is the problem according to Table 3.

   How test will be performed: It will be performed by IP Simulator, Pytest and Flake8.

5.2 Tests for Nonfunctional Requirements

5.2.1 Portability

Testing the portability of IP Simulator

1. T1: Portability

   Type: Manual

   Initial State: None.

   Requirement ID(s): NFR-1

   Input/Condition: Executes all tests in 5.1.1 and 5.1.2 in the IP Simulator in different OS, including Linux, Windows, MacOS.

   Output/Result: Successful test implies portability of the software.

   How test will be performed: The test will be performed manual by executing the software and tests in different OS, including Linux, Windows, MacOS. A successful test is specified by the verification of each test for functional requirements.

5.2.2 Usability

Testing the Usability of IP Simulator

1. T1: Usability

   Testing the usability will determine if the users have an efficient interaction with the software and also the software is easy to use or not. The system will be tested against a usability test and survey which the details are in the Appendix 7.

   Type: Manual

   Initial State: The IP Simulator runs.

   Requirement ID(s): NFR-2

   Input/Condition: Any tests considering inputs criteria can be run by a group of individuals who have varying levels of experience with inverted pendulum systems and the software, including at least an undergraduate Physics student, an undergraduate Engineering student, a high school student, and the domain expert. We will administer the questionnaire to the participants while they use the software and observe their behavior and note any issues they encounter.

   Output/Result: According to the feedbacks from the survey, the usability of the software will be tested.

   How test will be performed: Manually, as it is a test of usability by the user, and thus requires the test to be run by the users.

5.2.3 Accuracy

Testing the Accuracy of IP Simulator

1. T1: Accuracy

   The accuracy test should be done through comparing the results with a similar system which is available on the Internet and has been developed using MATLAB. [1]

   Type: Manual

   Initial State: The IP Simulator and the similar project in MATLAB with the same inputs run.

Requirement ID(s): NFR-3

Input/Condition: All of the test in 5.1.1 will be done.

Output/Result: Comparing the outputs of IP Simulator to the result of [1]. the relative error

How test will be performed: Manually, as both the projects should be run manually.

5.3 Traceability Between Test Cases and Requirements

A traceability between test cases and requirements is shown in Table 4.

Table 4: Traceability Matrix between tests and requirements

Test Cases	R1	R2	R3	R4	NFR1	NFR2	NFR3
5.1.1	X	X					
5.1.2			X	X			
5.2.1					X		
5.2.2						X	
5.2.3							X

6 Unit Test Description

6.1 Unit Testing Scope

This section is intentionally left blank until the MIS is completed.

References

[1] Huthaifa AL-Khazraji. Matlab code of inverted pendulum on cart using ode45 (animation), Apr 2022. URL https://www.youtube.com/watch?v=glf8HC5CiyE&t=162s.

[2] Andreas Hinderks, Martin Schrepp, and Jörg Thomaschewski. User experience questionnaire. URL https://www.ueq-online.org/.

[3] Swati Tawde. Code walkthrough. URL https://www.educba.com/code-walkthrough/.

7 Appendix

7.1 Usability Survey Questions?

To test usability of the IP Simulator, the Hinderks et al. questionnaire [2] will be used.