Project Title: System Verification and Validation Plan for Mechatronics

Team #20, OpenASL
Robert Zhu zhul49
Zifan Meng mengz17
Jiahui Chen chenj194
Kelvin Huynh huynhk12
Runze Zhu zhur25
Mirza Nafi Hasan hasanm21

November 02, 2022

1 Revision History

Date	Version	Notes
November 02, 2022	1.0 Everyone	
Date 2	1.1	Notes

Contents

1 Revision History **ii**

2 Symbols, Abbreviations and Acronyms **v**

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

4 Plan **1**
- 4.1 Verification and Validation Team . . . 1
- 4.2 SRS Verification Plan . . . 2
- 4.3 Design Verification Plan . . . 2
- 4.4 Implementation Verification Plan . . . 2
- 4.5 Automated Testing and Verification Tools . . . 2
- 4.6 Software Validation Plan . . . 2

5 System Test Description **2**
- 5.1 Tests for Functional Requirements . . . 2
  - 5.1.1 Area of Testing1 . . . 3
  - 5.1.2 Area of Testing2 . . . 3
- 5.2 Tests for Nonfunctional Requirements . . . 3
  - 5.2.1 Area of Testing1 . . . 4
  - 5.2.2 Area of Testing2 . . . 4
- 5.3 Traceability Between Test Cases and Requirements . . . 4

6 Unit Test Description **4**
- 6.1 Unit Testing Scope . . . 4
- 6.2 Tests for Functional Requirements . . . 5
  - 6.2.1 Module 1 . . . 5
  - 6.2.2 Module 2 . . . 5
- 6.3 Tests for Nonfunctional Requirements . . . 5
  - 6.3.1 Module ? . . . 6
  - 6.3.2 Module ? . . . 6
- 6.4 Traceability Between Test Cases and Modules . . . 6

7 Appendix **7**
- 7.1 Symbolic Parameters . . . 7
- 7.2 Usability Survey Questions? . . . 7

List of Tables

1 Symbols, Abbreviations, and Acronyms v
2 Verification and Validation Team Members and Roles 1

2 Symbols, Abbreviations and Acronyms

Term, Abbreviation, or Acronym	**Description**
ASL	Shorthand for American Sign Language. It is a form of sign language primarily used in the US and in parts of Canada
CFR	Shorthand for Camera Functional Requirement
CV	Shorthand for computer vision, computer vision is an interdisciplinary scientific field that deals with how computers can gain high-level understanding from digital images or videos
MLFR	Shorthand for Machine Learning Functional Requirement
NFR	Shorthand for Non-Functional Requirement
OpenCV	Shorthand for computer vision, computer vision is an interdisciplinary scientific field that deals with how computers can gain high-level understanding from digital images or videos
RDP	Shorthand for Real-time Data Processing
SRS	Shorthand for System Requirement Specification
TC	Shorthand for Test Case

Table 1: Symbols, Abbreviations, and Acronyms

3 General Information

3.1 Summary

[Say what software is being tested. Give its name and a brief overview of its general functions. —SS]

3.2 Objectives

[State what is intended to be accomplished. The objective will be around the qualities that are most important for your project. You might have something like: "build confidence in the software correctness," "demonstrate adequate usability." etc. You won't list all of the qualities, just those that are most important. —SS]

3.3 Relevant Documentation

The relevant documentation used to formulate the VnV plan include:

- SRS
- Hazard Analysis
- Development Plan

4 Plan

[Introduce this section. You can provide a roadmap of the sections to come. —SS]

4.1 Verification and Validation Team

Name	Responsibility
Robert Zhu	White/Black Box Testing; Manual SRS Verification
Zifan Meng	OpenCV Verification; Manual code Verification
Jiahui Chen	End-to-End Testing; Manual SRS Verification
Kelvin Huynh	Machine Learning Verification; Manual code Verification
Runze Zhu	White/Black Box Testing; End-to-End Testing
Mirza Nafi Hasan	Performance Testing; Manual code Verification
Classmate Peer Review	Provide peer reviews for our project
Dr. Spencer Smith / TAs	Provide reviews and feedback for our project

Table 2: Verification and Validation Team Members and Roles

4.2 SRS Verification Plan

[List any approaches you intend to use for SRS verification. This may just be ad hoc feedback from reviewers, like your classmates, or you may have something more rigorous/systematic in mind.. —SS]
[Remember you have an SRS checklist —SS]

4.3 Design Verification Plan

[Plans for design verification —SS]
[The review will include reviews by your classmates —SS]
[Remember you have MG and MIS checklists —SS]

4.4 Implementation Verification Plan

[You should at least point to the tests listed in this document and the unit testing plan. —SS]
[In this section you would also give any details of any plans for static verification of the implementation. Potential techniques include code walkthroughs, code inspection, static analyzers, etc. —SS]

4.5 Automated Testing and Verification Tools

[What tools are you using for automated testing. Likely a unit testing framework and maybe a profiling tool, like ValGrind. Other possible tools include a static analyzer, make, continuous integration tools, test coverage tools, etc. Explain your plans for summarizing code coverage metrics. Linters are another important class of tools. For the programming language you select, you should look at the available linters. There may also be tools that verify that coding standards have been respected, like flake9 for Python. —SS]
[The details of this section will likely evolve as you get closer to the implementation. —SS]

4.6 Software Validation Plan

[If there is any external data that can be used for validation, you should point to it here. If there are no plans for validation, you should state that here. —SS]

5 System Test Description

5.1 Tests for Functional Requirements

[Subsets of the tests may be in related, so this section is divided into different areas. If there are no identifiable subsets for the tests, this level of document structure can be removed. —SS]

[Include a blurb here to explain why the subsections below cover the requirements. References to the SRS would be good. —SS]

5.1.1 Area of Testing1

[It would be nice to have a blurb here to explain why the subsections below cover the requirements. References to the SRS would be good. If a section covers tests for input constraints, you should reference the data constraints table in the SRS. —SS]

Title for Test

1. test-id1
   Control: Manual versus Automatic

   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
   Control: Manual versus Automatic

   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:

5.1.2 Area of Testing2

...

5.2 Tests for Nonfunctional Requirements

[The nonfunctional requirements for accuracy will likely just reference the appropriate functional tests from above. The test cases should mention reporting the relative error for these tests. —SS]
[Tests related to usability could include conducting a usability test and survey. —SS]

5.2.1 Area of Testing1

Title for Test

1. test-id1
   Type:

   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:

5.2.2 Area of Testing2

...

5.3 Traceability Between Test Cases and Requirements

[Provide a table that shows which test cases are supporting which requirements. —SS]

6 Unit Test Description

[Reference your MIS and explain your overall philosophy for test case selection. —SS]
[This section should not be filled in until after the MIS has been completed. —SS]

6.1 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]

6.2 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]

6.2.1 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. ...

6.2.2 Module 2

...

6.3 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]

6.3.1 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:

6.3.2 Module ?

...

6.4 Traceability Between Test Cases and Modules

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

References

Author Author. System requirements specification. https://github.com/..., 2019.

7 Appendix

This is where you can place additional information.

7.1 Symbolic Parameters

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

7.2 Usability Survey Questions?

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