System Verification and Validation Plan for ImgBeamer

Joachim de Fourestier

February 17, 2023

1 Revision History

Date	Version	Notes
2023/02/17 1	1.0	First version
Date 2	1.1	Notes

Contents

List of Tables

[Remove this section if it isn't needed —SS]

List of Figures

[Remove this section if it isn't needed —SS]

2 Symbols, Abbreviations and Acronyms

symbol	description
SRS	Software Requirements Specification
T	Test
VnV	Verification and Validation

[symbols, abbreviations or acronyms — you can simply reference the SRS (de Fourestier, 2023d) tables, if appropriate —SS]

[Remove this section if it isn't needed —SS]

This document lays out the Verification and Validation (VnV) plan of the ImgBeamer (scanning electron microscope image formation) as described by the Software Requirements Specification (SRS) document (de Fourestier, 2023d). Testing of the software and its components is conducted to build confidence in its usability, accuracy, and ultimately whether it meets the SRS.

3 General Information

3.1 Summary

The ImgBeamer software aims to be a demonstration and learning tool of how a scanning electron microscope (SEM) formulates an image. The idea is to help visualize the influence trends of imaging parameters (as defined in the SRS) on image quality (or clarity) and resolution.

3.2 Objectives

The objective is to produce a tool that is easy to use (*usability*). It should feel intuitive to the user and should provide easy to understand information. This is so that the user may identify any trends in how the final image is affected with relative ease and confidence. As a secondary goal, the *accuracy* in the trends are important. As such, the image metrics should be straightforward to understand and provide some kind of assurance away from the subjective nature of image quality.

3.3 Relevant Documentation

There are multiple design documents that provide the important and intimate details to understand some of the concepts that are being tested. These documents are the following:

- Problem Statement (de Fourestier, 2023c)
- SRS (de Fourestier, 2023d)
- VnV Report (de Fourestier, 2023e)

4 Plan

In this section, multiple plans are described to test and inspect the software with an emphasis on *usability*. Multiple approaches and perspectives will be employed by the VnV team (4.1) to help build confidence in the requirements, to avoid any missed important details, and to deliver on the qualities as mentioned in the objective (3.2).

4.1 Verification and Validation Team

The members of the VnV team as well their individual roles are listed in the following table:

Role	Name
Project Supervisor	Dr. Spencer Smith
Author	Joachim de Fourestier
Domain Expert	Karen Wang
SRS Reviewer	Jason Balaci
VnV Plan Reviewer	Sam Crawford
MG + MIS Reviewer	Lesley Wheat
Expert Consultant	Dr. Nabil Bassim
Expert Consultant	Michael W. Phaneuf

Table 1: Table of the VnV Team Members

[Dr. Bassim is my PhD Supervisor, and Mr. Phaneuf is my co-supervisor (employer/industry), since I am doing an industrial PhD. Much of this project was originally conceptualized with the help and guidance of Mr. Phaneuf. —Author]

4.2 SRS Verification Plan

The SRS document is reviewed by the project supervisor, the SRS reviewer and the author. Some input may be given by the expert consultants if time permits. Most of the feedback has been provided as issues on GitHub, or as annotated documents, or by verbal exchange. The author is then expected

make the resolve the issues and makes accordingly throughout the development of the project. The key objectives are to verify that the software requirements and the documentation are coherent and sound to the intended audience as defined in the SRS.

4.3 Design Verification Plan

Much of the conceptualization was done after having multiple discussions with the expert consultants and having done literature review for preexisting tools and documentation relevant to the project. Decision are made with focus on the usability, with little to no setup being required. The design and implementation is documented in the MG / MIS (de Fourestier, 2023a,b). The VnV team as well as any volunteer users are welcome to provide their input (through GitHub issues). Eventually, the project may be published in a journal where the software ImgBeamerand its accompanying documentation will likely be rigorously reviewed.

[Not sure what was appropriate to put here. —Author]

4.4 Verification and Validation Plan Verification Plan

The goal is to uncover any mistakes and reveal any risks through the supervision and review of the VnV team members. Once most of the work has been done, the work and accompanying documentation shall undergo a final vetting process. Mainly, the team will check whether the documented testing plans and verification process have been accomplished and the requirements fulfilled.

4.5 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]

4.6 Automated Testing and Verification Tools

The image metric shall be unit-tested using pytest for automated testing of any algorithms implemented in Python and QUnit shall be used for those implemented in javascript. The unit tests are listed in section 6. As for linter, flake8 shall be used for Python code and ESLint for javascript code. The sewar python package will be used as a reference implementation in Python for the image quality metrics.

4.7 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]

[You might want to use review sessions with the stakeholder to check that the requirements document captures the right requirements. Maybe task based inspection? —SS]

[This section might reference back to the SRS verification section. —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 here. —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 here. 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. Not all projects will necessarily have nonfunctional requirements related to accuracy —SS]

[Tests related to usability could include conducting a usability test and survey. The survey will be in the Appendix. —SS]

[Static tests, review, inspections, and walkthroughs, will not follow the format for the tests given below. —SS]

5.2.1 Area of Testing1

Title for Test

1. test-id1

   Type: Functional, Dynamic, Manual, Static etc.

   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 (detailed design document) and explain your overall philosophy for test case selection. —SS] [This section should not be filled in until after the MIS (detailed design document) 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

J. de Fourestier. Module guide for imgbeamer, 2023a. URL https://github.com/joedf/CAS741_w23/blob/main/docs/Design/SoftArchitecture/MG.pdf.

J. de Fourestier. Module interface specification for imgbeamer, 2023b. URL https://github.com/joedf/CAS741_w23/blob/main/docs/Design/SoftDetailedDes/MIS.pdf.

J. de Fourestier. Problem statement and goals: Imgbeamer, 2023c. URL https://github.com/joedf/CAS741_w23/blob/main/docs/ProblemStatementAndGoals/ProblemStatement.pdf.

J. de Fourestier. Software requirements specification for imgbeamer: Scanning electron microscope image formation, 2023d. URL https://github.com/joedf/CAS741_w23/blob/main/docs/SRS/SRS.pdf.

J. de Fourestier. Verification and validation report: Imgbeamer, 2023e. URL https://github.com/joedf/CAS741_w23/blob/main/docs/VnVReport/VnVReport.pdf.

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]

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning. Please answer the following questions:

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning. Please answer the following questions:

1. What knowledge and skills will the team collectively need to acquire to successfully complete the verification and validation of your project? Examples of possible knowledge and skills include dynamic testing knowledge, static testing knowledge, specific tool usage etc. You should look to identify at least one item for each team member.

2. For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?