System Verification and Validation Plan for FBP CT Image Reconstruction

Qianlin Chen

February 24, 2025

Revision History

Date	Version	Notes
Feb 24, 2025	1.0	Initial VnV
...	...	...

Contents

List of Tables

1 Symbols, Abbreviations, and Acronyms

symbol	description
MG	Module Guide
MIS	Module Interface Specification
SRS	Software Requirement Specification
FBP	Filter Back Projection
TC	Test Case
VnV	Verification and Validation

For complete symbols used within the system, please refer the section 1 in SRS document.

This document outlines the VnV plan for FBP CT Image Reconstruction, ensuring compliance with functional and non-functional requirements through systematic verification and testing. It begins with General Information (Section 2), covering objectives and relevant documentation, followed by the VnV Plan (Section 3), which details team responsibilities, SRS and design verification, and automated testing tools. System Tests (Section 4) validate functional and non-functional requirements, including input and output tests, with traceability ensuring full requirement coverage. Lastly, Unit Test Description (Section 5) defines the testing scope, module-specific functional and non-functional tests, and the mapping of test cases to system components.

2 General Information

2.1 Summary

This document reviews the verification and validation plan for Filtered Back Projection (FBP) CT Image Reconstruction. The FBP method is used to reconstruct cross-sectional images from projection data collected in CT scans. The accuracy and performance of the reconstruction process are evaluated based on predefined validation criteria. The system's usability, maintainability, and reliability are also assessed using user inputs and experimental results.

2.2 Objectives

The purpose of the validation plan is to define how system validation will perform at the end of the project. The strategy will use to assess whether the developed system accomplishes the design goals. Also, the verification plan includes test strategies, definitions of what will be tested, and a test matrix with detailed mapping connecting the tests performed to the system requirements.

Due to resource constraints, validation will be limited to peer-based testing for accuracy and usability. While this ensures basic functionality and user experience evaluation, the absence of domain experts, such as doctors or researchers, prevents validation in real-world clinical or research applications.

2.3 Challenge Level and Extras

The challenge level for this project is classified as general, requiring a rigorous verification and validation process to ensure the accuracy and performance of the system. The validation plan includes extensive functional and non-functional testing, leveraging automated tools, formal proofs, and usability evaluations to assess correctness, efficiency, and maintainability.

Additionally, this project incorporates code walkthroughs, user documentation, and design reviews to enhance system reliability and comprehensibility. The scope of validation covers core algorithmic accuracy while excluding external dependencies, which are assumed to be pre-validated. This structured approach ensures that the system meets its defined performance, usability, and verification requirements while adhering to resource constraints.

2.4 Relevant Documentation

The relevant documentation for the VnV includes Problem Statement which identifies the proposed idea, System Requirements Specifications which provides information about the requirement of the proposed system, VnV Report for validation and verification, MG and MIS design documents for unit testing (found in Github Repository).

3 Plan

This section describes the verification and validation plan for the FBP CT Image Reconstruction.The planning starts with the verification and validation team in Section 3.1, followed by the SRS verification plan (Section 3.2), design verification plan (Section 3.3), verification and validation verification plan (Section 3.4), implementation verification plan (Section 3.5), Automated testing and verification tools (Section 3.6), and Software validation plan (Section 3.7).

3.1 Verification and Validation Team

Name	Document	Role	Description
Dr. Spencer Smith	All	Instructor/ Reviewer	Review the documents, design and documentation style.
Qianlin Chen	All	Author	Create and manage all documentation, develop the VnV plan, conduct VnV testing, and verify the implementation.
Xunzhou Ye	All	Domain Expert Reviewer	Review all the documents.

Table 1: Verification and validation team

3.2 SRS Verification Plan

The SRS document for FBP CT Image Reconstructionwill be verified using a structured review process. The verification plan consists of the following steps:

Initial Review

Assigned reviewers, instructor (Dr. Smith) and domain expert (Xunzhou). will conduct a manual review of the SRS document. In this step, a structured SRS Checklist will be used to systematically evaluate key aspects such as requirement completeness, consistency, feasibility, and traceability.

Issue Tracking

eviewers will provide feedback by documenting identified issues in an issue tracker (GitHub Issues). Each issue will be assigned to the document owner for later revision.

Revision

The document owner will address reported issues by making necessary modifications to the SRS. Any unresolved or disputed concerns will be discussed among the reviewers.

Final Documentation

After revisions, a final review will be conducted to confirm that all identified issues have been adequately addressed.

3.3 Design Verification Plan

The design documentation, including the Module Guide (MG) and Module Interface Specification (MIS), will be reviewed to ensure accuracy and completeness. The verification process will involve a static analysis approach, where assigned reviewers will inspect the documents to confirm that they align with system requirements and architectural design principles. Reviewers will provide feedback through an issue-tracking system, allowing the document owner to address concerns systematically.
A structured MG Checklist and MIS Checklist will be used to ensure consistency, correctness, and traceability. The finalized documents will be reviewed again to validate that all identified issues have been resolved.

3.4 Verification and Validation Plan Verification Plan

The review process for Verification and Validation (VnV) Plan will involve assigned team members in Table 1 conducting a structured inspection to verify that the plan aligns with project requirements and verification strategies. Identified issues will be documented in an issue-tracking system for resolution.
A VnV Checklist will be used to assess key aspects of the plan, including coverage of testing methodologies and traceability of requirements. The final review will confirm that all necessary corrections have been implemented before approval.

3.5 Implementation Verification Plan

- Static Verification:
  - Code Walkthrough: A structured review process where team members manually inspect the code for correctness, maintainability, and adherence to design specifications. This process ensures that critical components meet functional requirements before execution.
- Dynamic Verification:

- Unit Testing: All test cases outlined in Section 5 will be executed, covering both functional and non-functional requirements. These tests ensure correctness at the module and system levels, validating reconstruction accuracy and computational efficiency.

3.6 Automated Testing and Verification Tools

To ensure the correctness and efficiency of the FBP CT Image Reconstruction, the following automated testing and verification tools will be utilized:

Unit Testing
pytest will be used to test individual functions, such as backprojection and filtering operations, to verify that they produce the expected output given test input data.

Performance Analysis
cProfill will analyze execution time and identify performance bottlenecks in computationally intensive functions like fourier transforms and interpolation.

Static Analysis
flake8 will enforce python code standards, ensuring maintainability and readability.

Test Coverage
coverage.py will measure test coverage to ensure that all critical functions are tested and validated against various datasets.

Visualization
matplotlib will be used to visualize sinograms and reconstructed images, allowing for manual verification of reconstruction accuracy.

Continuous Integration (CI) tools will be Github Action. Tools for unit testing, static analysis can run automatically.

3.7 Software Validation Plan

Currently, there are no external datasets or domain experts available to validate the FBP CT Image Reconstruction system. Since the system is not yet

in real-world use and lacks access to researcher for review, formal validation cannot be conducted at this stage.

However, internal review sessions may be used to ensure that the system aligns with specified requirements. Future validation efforts could include structured user testing or a Rev 0 demo if external stakeholders become available. Until then, verification efforts will focus on functional correctness through testing, as outlined in the SRS verification section.

4 System Tests

This section outlines the testing framework for verifying the system, covering both functional and non-functional requirements. Section 4.1 details functional tests, including input tests (4.1.2) to validate data preprocessing and output tests (4.1.1) to assess reconstructed image accuracy. Section 4.2 focuses on non-functional testing, such as performance and usability evaluations. Finally, Section 4.3 ensures traceability between test cases and system requirements, maintaining comprehensive coverage and validation of the system's expected behavior.

4.1 Tests for Functional Requirements

The functional requirements in SRS are divided into input and output tests. R1 and R2 correspond to input-related tests, ensuring that the system correctly processes input images, projection data, and transformations. The tests for R1 and R2 are in the Section 4.1.1
R3 correspond to output-related tests, verifying the correctness of Fourier filtering, back-projection, and reconstructed attenuation values. The test for the R3 is in the Section 4.1.2

4.1.1 Input Tests

Test for Input - 2D grayscale image

1. test-input-image- id1

   Control: Automatic

   Initial State: Pending Input.

Input:

- A 2-D array represents the intensity measurements in different projection angles. Sample data input can be found in Cancer Image Archive under the title 'Images Phantom Object Only'.
- The projection angles in degrees. In this test, a range of 180 angles from 0 to 179 degrees is used.

Output: A 2-D array represents the Sinogram. The result for the input data can be found in Cancer Image Archive under the title 'Clinical Data'.

Test Case Derivation: This test ensures that the system correctly accepts a valid grayscale 2D input image, applies the Radon Transform, and produces a valid sinogram matrix.
The system should return an output that maintains consistency in shape and correctly reflects the values after applying the Radon Transform.

How test will be performed: The test will be automated using PyTest.

Test for Input - filter type

1. test-input-filter-type- id2

   Control: Automatic

   Initial State: System has generated the sinogram matrix and the user is ready to select the ramp filter.

   Input: Shepp-logan filter type and the size of sinogram in the Table2.

   Output: A 1-D array that represent the selected Fourier filter in the Table2.

   Test Case Derivation:
   This test ensures that the system correctly processes user-selected filter types.
   A valid ramp filter should be applied as expected, modifying the sinogram by filtering out the low frequency.

   How test will be performed: The test will be automated using PyTest.

1. test-input-filter-type- id3

   Control: Automatic

   Initial State: System has generated the sinogram matrix and the user is ready to select the ramp filter.

   Input: Shepp-logan filter type and the size of sinogram in the Table2.

   Output: A 1-D array that represent the selected Fourier filter in the Table2.

   Test Case Derivation:
   This test ensures that the system correctly processes user-selected filter types.
   A valid shepp-logan filter should be applied as expected, modifying the sinogram by filtering out the high frequency.

   How test will be performed: The test will be automated using PyTest.

1. test-input-invalid-filter-type- id4

   Control: Automatic

   Initial State: System has generated the sinogram matrix and the user is ready to select the ramp filter.

   Input: No filter type and the size of sinogram in the Table2.

   Output: A 1-D array that represent the selected Fourier filter in the Table2.

   Test Case Derivation:
   This test ensures that the system correctly processes user-selected filter types.
   A valid shepp-logan filter should be applied as expected, modifying the sinogram by filtering out the high frequency.

   How test will be performed: The test will be automated using PyTest.

Test Id	Input	Expected Output
test-input-filter-type-id2	Filter Type: ramp (str); Sinogram size: 10 (int)	1-D array represent ramp filter variable: [0.0, 0.314159, 0.628318, 0.942477, 1.256636, 1.570796, 1.256636, 0.942477, 0.628318, 0.314159]
test-input-filter-type-id3	Filter Type: shepp (str); Sinogram size: 10 (int)	1-D array represent shepp-logan filter variable: [0.0, 0.2984, 0.5630, 0.7958, 0.9817, 1.0986, 0.9817, 0.7958, 0.5630, 0.2984]
test-input-invalid-filter-type-id4	anything other than ramp and shepp	None

Table 2: The inputs and outputs for filter tests

4.1.2 Output Tests

Test for Output - attenuation

1. test-output-attenuation- id5

   Control: Automatic

   Initial State: The system is ready to perform back-projection on a given sinogram and filter type.

   Input: A sinogram and the ramp filter in the Table 3.

   Output: A 2-D array that represent the reconstructed attenuation in the Table3.

   Test Case Derivation: This test ensures that the system correctly applies the back-projection process to reconstruct an image from a given sinogram. The test verifies that the output maintains the expected shape and that the reconstructed attenuation values correctly represent the original structure.

   How test will be performed: The test will be automated using PyTest.

1. test-output-attenuation- id6

   Control: Automatic

   Initial State: The system is ready to perform back-projection on a given sinogram and filter type.

   Input: A sinogram and the shepp-logan filter in the Table 3.

   Output: A 2-D array that represent the reconstructed attenuation in the Table3.

   Test Case Derivation: This test ensures that the system correctly applies the back-projection process to reconstruct an image from a given sinogram. The test verifies that the output maintains the expected shape and that the reconstructed attenuation values correctly represent the original structure.

   How test will be performed: The test will be automated using PyTest.

Test Id	Input	Expected Output
test-output-attenuation-id5	Sinogram can be found in Cancer Image Archive under the title 'Clinical Data'; Filter Type: ramp (str); Output size: 256	Attenuation can be found in Cancer Image Archive under the title 'Patients Report'
test-output-attenuation-id6	Sinogram can be found in Cancer Image Archive under the title 'Clinical Data'; Filter Type: shepp (str); Output size: 256	Attenuation can be found in Cancer Image Archive under the title 'Patients Report'

Table 3: The inputs and outputs for FBP tests

4.2 Tests for Nonfunctional Requirements

Nonfunctional requirements for FBP system are given in SRS Section 5.2.

Test for Accuracy

1. test-accuracy- id7

   Type: Functional, Dynamic.

   Initial State: Reconstructed image is available.

   Input/Condition: The reconstructed image and the corresponding ground truth image.

   Output/Result: Error margin (Mean Square Error) is computed and compared against the predefined tolerance level.

   How test will be performed: The test will compare the reconstructed images with ground-truth images using metrics like Mean Squared Error (MSE). The relative error for each test case will be recorded and analyzed.

Test for Usability

1. test-usability- id8

   Type: Manual.

   Initial State: System interface is implemented. Usability survey in Section 6.2 is predefined.

   Input/Condition: Joe in Table 1 perform tasks follows the survey.

   Output/Result: Feedback from usability survey, including ease of use, clarity of instructions, and efficiency of interaction.

   How test will be performed: A usability study will be conducted where users perform predefined tasks. Their experience will be recorded, and a survey will be provided to assess ease of use, with results summarized in a report.

Test for Maintainability

1. test-maintainability- id9

   Type: Manual, Static.

Initial State: System implementation is available.

How test will be performed:
To ensure the maintainability of the system, a structured code review process will be conducted. The review will focus on the clarity of documentation, ease of understanding the code structure, and the effort required for future modifications or enhancements.

Additionally, maintainability will be assessed by tracking modification times for common updates, such as adding new filters or adjusting reconstruction parameters. If changes require excessive refactoring, recommendations for improving the code's modularity will be documented.

The review will be conducted collaboratively, involving the assigned document reviewers listed in Table 1. The findings will be summarized in a Github Issue, outlining strengths and areas for improvement.

Test for Portability

1. test-portability- id10

   Type: Functional, Dynamic.

   Initial State: The software is fully implemented.

   Input/Condition: Different operating systems (Windows, macOS, Linux).

   Output/Result: Verification that the system runs without errors across all tested platforms.

   How test will be performed: The software will be deployed and executed on multiple operating systems, ensuring it performs consistently without requiring additional setup.

Test for Reusability

1. test-reusability- id11

   Type: Manual.

   Initial State: API is implemented and documented.

Input/Condition: External developers in Table 1 integrating a new filter beyond Ramp and Shepp-Logan.

Output/Result: Confirmation that new filters can be added with minimal effort using the provided API.

How test will be performed: Developers will attempt to add a new filter, and the time/effort required will be assessed. Documentation clarity and API flexibility will also be evaluated.

4.3 Traceability Between Test Cases and Requirements

test \ requirement	R1	R2	R3	NFR1	NFR2	NFR3	NFR4	NFR5
id1	X							
id2		X						
id3		X						
id4		X						
id5			X					
id6			X					
id7				X				
id8					X			
id9						X		
id10							X	
id11								X

Table 4: Traceability Matrix Showing the Connections Between Requirements and Test Cases

5 Unit Test Description

[This section should not be filled in until after the MIS (detailed design document) has been completed. —SS]

[Reference your MIS (detailed design document) and explain your overall philosophy for test case selection. —SS]

[To save space and time, it may be an option to provide less detail in this section. For the unit tests you can potentially layout your testing strategy here. That is, you can explain how tests will be selected for each module. For instance, your test building approach could be test cases for each access program, including one test for normal behaviour and as many tests as needed for edge cases. Rather than create the details of the input and output here, you could point to the unit testing code. For this to work, you code needs to be well-documented, with meaningful names for all of the tests. —SS]

5.1 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]

5.2 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]

5.2.1 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. ...

5.2.2 Module 2

...

5.3 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]

5.3.1 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:

5.3.2 Module ?

...

5.4 Traceability Between Test Cases and Modules

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

6 Appendix

This is where you can place additional information.

6.1 Symbolic Parameters

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

6.2 Sample Usability Survey Questions

Thank you for participating in this usability survey. Your feedback will help us improve the system's usability and user experience.

Participant Information:

- Role: _____ (e.g., Researcher, Student, Developer)
- Experience with similar tools: _____ (None / Beginner / Intermediate / Expert)
- Operating system used: _____ (Windows / macOS / Linux)

Task Performance: On a scale from 1 (very difficult) to 5 (very easy), rate how easy it was to complete the following tasks:

Task	1	2	3	4	5
Loading projection data					
Selecting a filter					
Generating a sinogram					
Applying back-projection					
Viewing and comparing images					

System Usability: Please rate the following aspects of the system on a scale from 1 (strongly disagree) to 5 (strongly agree).

Statement	1	2	3	4	5
The system was easy to use.					
The instructions were clear.					
I was able to complete tasks efficiently.					
The system's response time was acceptable.					
I would use this system again.					

Open-Ended Feedback:

- What did you like most about the system?

  _____

- What difficulties did you encounter?

  _____

- Any additional comments or suggestions?

  _____

Submission: Please submit your responses via email to `research@domain.com` or upload them to the project repository.

Thank you for your participation!