

LUNAR IMAGE SIMULATION AND VISUALIZATION

Report submitted for completion of training

(Satellite Meteorology and Oceanography Research and Training)

Programme of Space Applications Centre

**Submitted by**

**HARSHALKUMAR SUNILKUMAR PATEL**

**(Registration No. RS02034)**

B.Tech. (Computer Science and Engineering)

**Under the guidance of**

**Sri. K.SURESH**

SCI/ENG – SG

PSPD/PMPG/SIPA

Space Applications Centre(SAC) , ISRO Ahmedabad

**Institution**

**PARUL University**

Vadodara– 391760

Gujarat



**SRTD-RTMG-MISA**

**S**pace **A**pplications **C**entre (SAC) , ISRO

Ahmedabad, Guajrat

09 December 2024 to 12 April 2025

**वैज्ञानिक अनुसंधान एवं प्रशिक्षण प्रभाग (एसआरटीडी)**

**Scientific Research and Training Division (SRTD)**

**आरटीएमजी/मीसा/सैक RTMG/MISA/SAC**

**CERTIFICATE**

This is to certify that **Mr. Harshalkumar Sunilkumar Patel**, a student of B.Tech (Computer Science and Engineering) of Parul University, Vadodara, Gujarat has completed a four months (09 Dec-2024 to 12 April-2025) project on **"Lunar Image simulation and Visualization"** under the supervision of K.Suresh, Scientist-SG, PSPD/PMPG /SIPA, Space Applications Centre (SAC) , ISRO Ahmedabad. The research work was carried out through Scientific Research and Training Division (SRTD) of Space Applications Centre, Ahmedabad.

Acknowledgements

I would like to express my sincere gratitude to SAC ISRO, Ahmedabad for providing me with the opportunity to work on this project, ” Lunar Image Simulation and Visualization.” This experience has been invaluable in enhancing my knowledge of satellite imaging and digital image processing.

I extend my heartfelt thanks to my industry mentor, Mr. K. Suresh, for his continuous guidance, encouragement, and technical insights that helped shape this project. His expertise in satellite imaging and simulation provided me with a strong foundation to develop this system.

Lastly, I would like to thank my colleagues, peers, and everyone who supported and motivated me throughout this journey. This project would not have been possible without their constant encouragement and collaboration.

Harshalkumar SunilKumar Patel AI and AIDS, PIET

Parul University, Vadodara

Abstract

The Image Simulation and Visualization project aims to assist in designing and evaluating satellite imaging systems by simulating how a camera would capture images in a lunar environment. By allowing users to define various camera parameters such as position, altitude, roll, yaw, pitch, and lens settings, this system generates simulated images that closely represent the expected results from actual lunar missions.

The project is developed using PyQt5 and Qt Designer for the graphical user interface, while NumPy, PyQtGraph, and GDAL handle image processing and spatial data operations. The iterative waterfall model is followed to ensure systematic development and refinement based on feedback at each stage.

This system plays a crucial role in optimizing camera settings before deployment on lunar satellites, reducing experimental costs and improving mission accuracy. The feasibility and effectiveness of the simulation ensure that future moon exploration missions can be planned with greater precision and confidence.

TABLE OF CONTENTS

Table of Contents

Acknowledgements vi

Abstract vii

1 Introduction to Project 8

1.1 Project Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

1.2 Purpose . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

1.3 Objective . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

1.4 Scope (What it Can and Can’t Do) . . . . . . . . . . . . . . . . . . . . . . . . . . 9

1.5 Technology . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9

1.6 Project Planning . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10

1.6.1 Project Development Approach and Justification . . . . . . . . . . . . . . 10

1.6.2 Project Effort and Time, Cost Estimation . . . . . . . . . . . . . . . . . . 10

2 System Analysis 12

2.1 Study of Current System . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12

2.2 Requirements of New System . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12

2.3 System Feasibility . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13

2.3.1 Does the system contribute to the overall objectives of the organization? . . 13

2.3.2 Can the system be implemented using the current technology and schedule

constraints? . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13

2.3.3 Can the system be integrated with other systems already in place? . . . . . 13

2.4 Process in Proposed System . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13

2.5 Features of Proposed System . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14

2.6 List of Main Modules / Components / Processes / Techniques of New System /

Proposed System . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14

V

2.7 Selection of Hardware /Software /Algorithms /Methodology /Techniques . . . . 14

2.7.1 Hardware: . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14

2.7.2 Software: . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15

2.7.3 Algorithms: . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15

2.7.4 Methodology: . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15

2.7.5 Techniques: . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15

3 System Design 16

3.1 System Design & Methodology . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

3.1.1 Graphical User Interface (GUI) . . . . . . . . . . . . . . . . . . . . . . . 16

3.1.2 Camera Parameters Handling . . . . . . . . . . . . . . . . . . . . . . . . 16

3.1.3 Marker-Based Simulation . . . . . . . . . . . . . . . . . . . . . . . . . . 16

3.1.4 Footprint Preview . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

3.1.5 Image Processing . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

3.2 Data Structure /Process /Structure Design . . . . . . . . . . . . . . . . . . . . . . 17

3.2.1 Data Structure Design . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

3.2.2 Process Design . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

3.3 Input /Output and Interface Design . . . . . . . . . . . . . . . . . . . . . . . . . 18

3.3.1 State Transition Diagram (Optional) . . . . . . . . . . . . . . . . . . . . . 18

3.3.2 Samples of Forms, Reports, and Interface

3.3.3 Access Control /Security (If Applicable)

. . . . . . . . . . . . . . . . . . 18

. . . . . . . . . . . . . . . . . . 18

4 IMPLEMENTATION 19

4.1 Implementation Platform /Environment . . . . . . . . . . . . . . . . . . . . . . . 19

4.2 Process /Program /Technology /Modules Specification(s) . . . . . . . . . . . . . 19

4.2.1 Process Flow . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19

4.2.2 Technologies & Modules Used . . . . . . . . . . . . . . . . . . . . . . . . 20

4.2.3 Findings /Results /Outcomes . . . . . . . . . . . . . . . . . . . . . . . . 20

VI

TABLE OF CONTENTS

4.2.4 Result Analysis /Comparison /Deliberations . . . . . . . . . . . . . . . . 21

4.2.5 Implementation images : . . . . . . . . . . . . . . . . . . . . . . . . . . . 21

5 Testing 30

5.1 Testing Plan /Strategy . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 30

5.1.1 Testing Types Used . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 30

5.1.2 Testing Environment . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 30

5.2 Test Results and Analysis . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 30

5.2.1 Test Cases . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 31

5.3 Analysis of Test Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 31

6 Conclusion and Summary 32

6.1 Conclusion and Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 32

6.2 Limitations . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 32

6.3 Future Work . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33

vii PI

List of Tables

5.1 Test Cases for Image Simulation System . . . . . . . . . . . . . . . . . . . . . . . 31

viii

1 INTRODUCTION TO PROJECT

1.1 PROJE CT SUMMARY

This project focuses on Moon Surface Image Simulation using satellite imaging technologies to

simulate and visualize images of the moon’s surface. The goal is to understand how a camera will

capture an image in an unknown environment, such as the moon’s surface, and help in designing

and setting up camera parameters for such simulations. The project aims to provide an effective

system for simulating images by adjusting parameters like camera position, altitude, roll, yaw, and

pitch, which can be used in designing cameras for future lunar missions.

By simulating the moon’s surface images based on user input, the system can assist in camera

parameter optimization for space missions. It provides an interface where users can input pre-

captured images and define specific parameters, and in return, they will get a simulated moon

surface image.

1.2 PURPOSE

The primary purpose of this project is to simulate and visualize how the moon’s surface will appear

based on different camera parameters such as position, altitude, and orientation. The system will

serve as a tool for space scientists and engineers involved in the design and calibration of lunar

imaging systems.

By offering an interactive platform, the project aims to:

• Provide a realistic simulation of moon surface images based on camera position and settings.

• Aid in studying how camera will capture the images.

1.3 OBJECT IVE

The main objectives of this project are:

• To design a system that can take user input images (pre-captured) and simulate the appearance

of the moon’s surface under different camera parameters.

8

CHAPTER 1. INTRODUCTION TO PROJECT

• To help in studying for lunar missions by visualizing how the camera would capture images

of the moon from different perspectives.

• To develop an intuitive user interface using PyQt5 that allows users to interact with the

simulation and adjust parameters like latitude, longitude, altitude, and camera orientation.

1.4 SCOPE (WHAT IT CAN AND CAN’T DO)

Scope of the Project:

• The system can simulate images of the moon’s surface, enabling users to visualize how

different camera parameters impact the final image.

• The tool will allow users to input camera settings, such as camera position, altitude, and

orientation parameters, and generate simulated images of the lunar surface.

What the Project Can’t Do:

• It cannot provide real-time image capturing or real-world data from lunar surface cameras.

• It cannot generate new images of the moon based on raw data but will only work with

pre-captured input images.

• The simulation is limited to a static model and does not include dynamic changes (e.g.,

changing lunar phases or real-time environmental variations on the moon’s surface).

1.5 TECHNOLOGY

The project uses the following technologies:

• PyQt5: For creating the graphical user interface (GUI) that allows users to input and

manipulate parameters.

• PyQtGraph: Used for visualization, it provides a fast and efficient way to display simulated

images and graphical representations of camera settings.

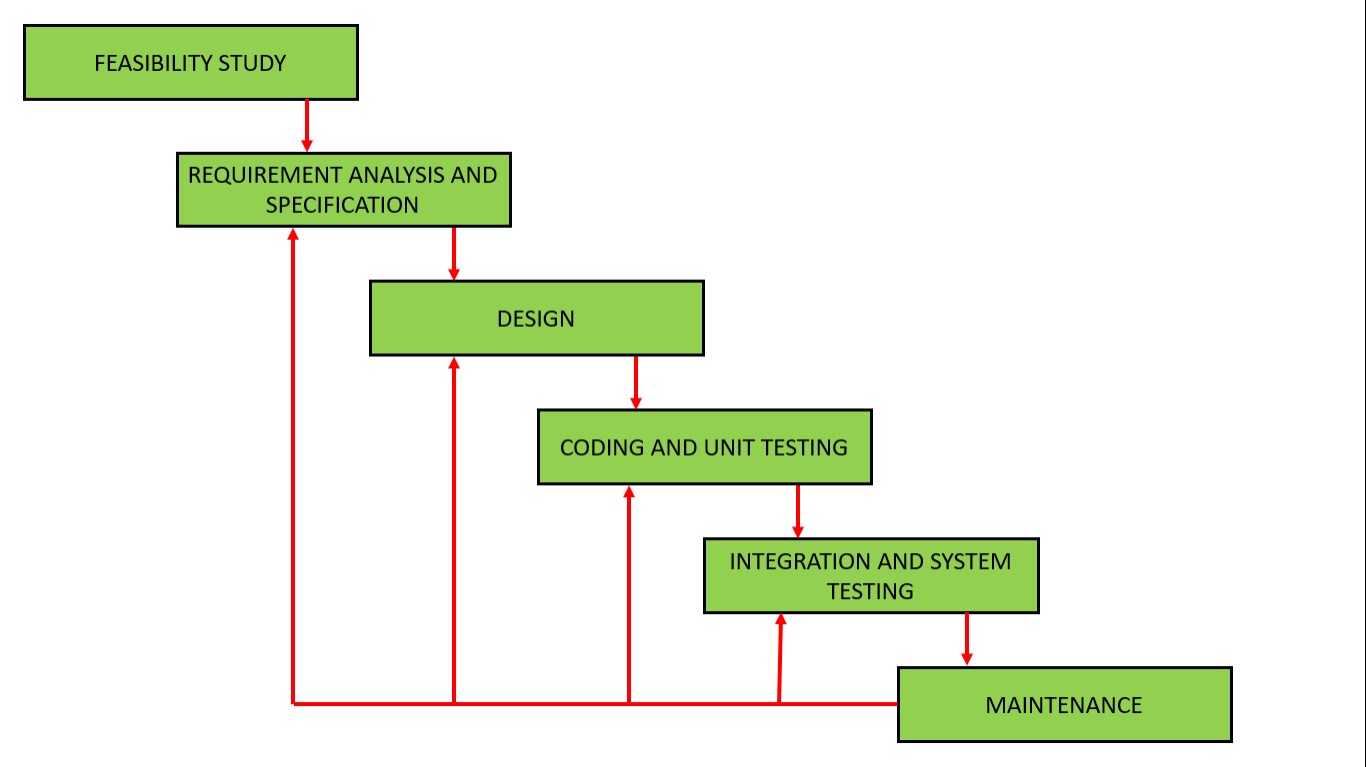
• GDAL: A library used to handle geographic data and transform spatial data for accurate

simulation of images.

• NumPy: Essential for performing mathematical operations and data transformations involved

in the image simulation.

9

CHAPTER 1. INTRODUCTION TO PROJECT

1.6 PROJE CT PLANNING

1.6.1 Project Development Approach and Justification

Figure 1.1: Iterative Waterfall Model

This project follows the Iterative Waterfall Model, allowing for incremental development and

continuous feedback after each phase. The iterative model provides flexibility to refine the design

after each cycle, ensuring that user needs are met and issues are addressed promptly.

Phase 1: Requirements Gathering and Initial Design: In this phase, project requirements

were gathered from experts in space imaging.

Phase 2: Prototype Development and User Interface Design: Based on the requirements,

an initial prototype of the GUI was developed using PyQt5. Early simulations with predefined

parameters were conducted to verify the system’s functionality.

Phase 3: Iteration and Refinement: After initial feedback from users, the system was

iteratively refined. New features like parameter adjustments and image enhancement tools were

added based on user input.

1.6.2 Project Effort and Time, Cost Estimation

The project is expected to take 3-4 months for completion. This includes time for prototype

development, testing, and refinement of the system.

Time Estimation:

• Prototype development: 1 month

• Testing and refinement: 2 months

10

CHAPTER 1. INTRODUCTION TO PROJECT

• Final adjustments and presentation: 1 month

Cost Estimation: The project will require already available hardware for testing.

11

2 SYSTEM ANALYSIS

2.1 STUDY OF CURRENT SYSTEM

Currently, there is no integrated system that allows users to simulate the appearance of the moon’s

surface based on varying camera parameters in a user-friendly way. Various image processing and

satellite simulation tools exist but often focus on post-processing images or limited camera settings.

Therefore, a need exists for a customizable, efficient tool that can provide simulations of lunar

surface images with the flexibility to adjust camera positions, orientations, and other parameters.

Existing systems for space imaging simulations may lack features like interactive user inputs

or adjustments for environmental factors such as altitude, yaw, and pitch. They also tend to be

rigid in their design, focusing mainly on theoretical models rather than offering detailed user-driven

simulations.

2.2 REQUIREMENTS OF NEW SYSTEM

The new system must fulfill the following requirements:

1. Customizable Parameters: The system should allow users to input and modify camera

parameters, such as altitude, latitude, longitude, pitch, yaw, and roll.

2. User-Friendly Interface: The system must have an intuitive graphical user interface (GUI)

developed using PyQt5, enabling easy input of parameters and viewing of simulated results.

3. Realistic Simulations: It should simulate realistic moon surface images using pre-captured

satellite images.

4. Support for Lunar Mission Data: The system must allow integration with data from lunar

missions such as Chandrayaan, enabling the simulation of different lunar regions and testing

of different camera configurations.

12

CHAPTER 2. SYSTEM ANALYSIS

2.3 SYSTEM FEAS IB ILITY

2.3.1 Does the system contribute to the overall objectives of the organization?

Yes, the system contributes to the overall objectives of ISRO, especially in the area of satellite

imaging and space exploration. By helping engineers design and set up optimal camera parameters

for lunar missions, the tool supports space mission readiness and camera configuration optimization.

It aligns with the vision of advancing space technology and enhancing satellite-based remote sensing

for lunar exploration.

2.3.2 Can the system be implemented using the current technology and schedule constraints?

Yes, the system can be implemented using current technologies such as PyQt5, NumPy, PyQtGraph,

and GDAL. These tools are highly capable for developing the GUI, performing mathematical

computations, and processing satellite image data. Additionally, the iterative waterfall model allows

for efficient development within the time constraints, as each phase will build upon feedback from

the previous iteration.

2.3.3 Can the system be integrated with other systems already in place?

Yes, the system is designed to integrate with existing simulation modules

2.4 PROCESS IN PROPOSED SYSTEM

The proposed system will include the following activities:

• Input Data: Users will provide pre-captured lunar images and set parameters such as latitude,

longitude, altitude, camera orientation, etc.

• Image Simulation: Based on the input parameters, the system will simulate a new image

of the moon’s surface, reflecting how the camera would capture the scene in those specific

parameters.

• Visualization: Simulated images will be visualized on a graphical interface, and users can

interact with the interface to adjust parameters.

• Export: After running the simulation, users can export the generated images or data for

further analysis.

13

CHAPTER 2. SYSTEM ANALYSIS

2.5 FEATURES OF PROPOSED SYSTEM

• Interactive User Interface: Developed using PyQt5, allowing users to easily input and

modify camera parameters and view simulated images.

• Simulation: Based on pre-defined camera parameters and the moon’s surface data.

• Parameter Control: Includes controls for various camera settings (e.g., latitude, longitude,

altitude, pitch, yaw, roll).

• Export and Report Generation: Users can save simulated images and generate detailed

reports for mission planning.

2.6 L I S T OF MAIN MODULES / COMPONENTS / PROCESSES /

TECHNIQUES OF NEW SYSTEM /PROPOSED SYSTEM

The main modules/components of the system will include:

• User Interface Module: Developed with PyQt5 for user input and interaction.

• Simulation: Handles image processing and simulation based on camera parameters using

NumPy and GDAL.

• Visualization Module: Uses PyQtGraph to display the simulated image and allow interactive

viewing.

• Data Integration Module: Interfaces with satellite data for real-time simulation.

• Export/Reporting Module: Allows users to export simulated images for analysis.

2.7 SELECTION OF HARDWARE / SOFTWARE / ALGORITHMS /

METHODOLOGY /TECHNIQUES

2.7.1 Hardware:

The project will be developed on standard computers with sufficient computational power to handle

image processing. Systems should have good graphic processing capabilities for efficient simulation

and rendering.

14

CHAPTER 2. SYSTEM ANALYSIS

2.7.2 Software:

• PyQt5 for GUI development

• NumPy for mathematical operations

• PyQtGraph for image visualization

• GDAL for handling geospatial data

2.7.3 Algorithms:

Parallel processing algorithms were implemented for fast processing.

2.7.4 Methodology:

The project will follow the Iterative Waterfall Model, ensuring that each phase undergoes continuous

refinement through feedback loops.

2.7.5 Techniques:

The system will employ camera parameter optimization and image simulation techniques to provide

users with a realistic simulation of lunar surface imagery.

15

3 SYSTEM DESIGN

3.1 SYSTEM DESIGN & METHODOLOGY

The image simulation system is designed to take a GeoTIFF image as input and simulate how

a camera would capture it in an unknown environment. The methodology follows a structured

pipeline consisting of image loading, parameter configuration, marker placement, and simulation

output. it’s made through Iterative Waterfall Model.

The key components of the system design are:

3.1.1 Graphical User Interface (GUI)

Built using PyQt and PyQtGraph, providing an interactive environment for parameter adjustments

and image visualization.

3.1.2 Camera Parameters Handling

Users can set parameters such as detector height, width, FOV, focal length, and altitude to define

the camera specifications.

3.1.3 Marker-Based Simulation

Users can place markers using latitude-longitude coordinates to define the simulation area. Single-

point mode allows one marker, while multi-point mode enables simulation between two points.

3.1.4 Footprint Preview

The system provides a footprint visualization before actual simulation to help users understand the

expected output coverage.

3.1.5 Image Processing

The core simulation logic processes the input GeoTIFF based on selected camera parameters,

generating the simulated image.

This structured approach ensures that the system effectively meets the objective of simulating

camera behavior for testing and design purposes.

16

CHAPTER 3. SYSTEM DESIGN

3.2 DATA STRUCTURE /PROCESS /STRUCTURE DESIGN

3.2.1 Data Structure Design

The system primarily works with the following data structures:

• Camera Parameter Dictionary: Stores values such as detector size, FOV, focal length, and

altitude.

• GeoTIFF Image Data: Handled as multi-dimensional arrays for processing.

• Marker Data: Stores latitude-longitude pairs for single and multi-point modes.

3.2.2 Process Design

The image simulation follows these steps:

1. Load Image: User selects a GeoTIFF image.

2. Set Camera Parameters: Parameters are either manually entered or selected from the

preloaded camera types via a combo box.

3. Mark Position(s): User marks a single or two-point location using the mouse or by entering

coordinates.

4. Preview Footprint: The system calculates and displays the expected coverage.

5. Simulate Image: The system processes the image according to camera parameters and outputs

the simulated image.

6. Save or Clear: Users can save results or reset parameters and markers.

17

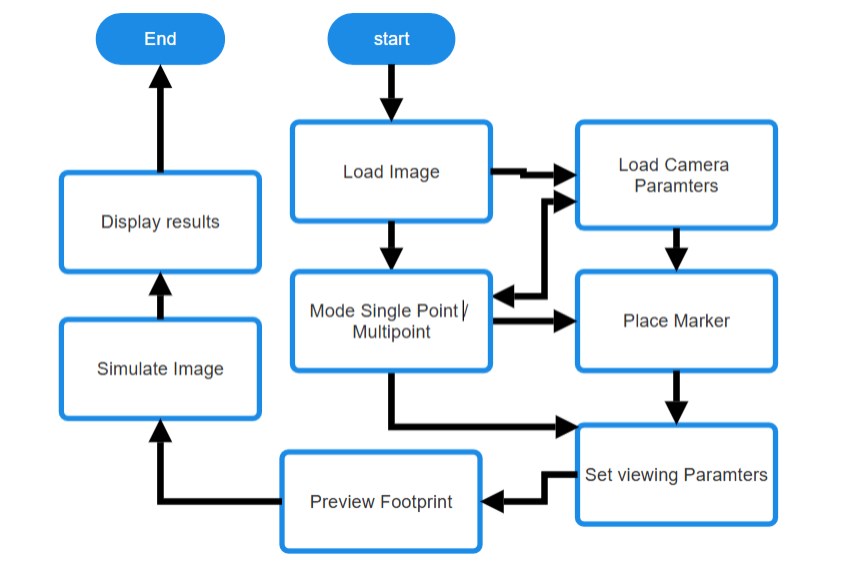
CHAPTER 3. SYSTEM DESIGN

Figure 3.1: User Flow Chart

3.3 INPUT /OUTPUT AND INTERFACE DESIGN

3.3.1 State Transition

State transition depict the various states of the system,

such as:

Idle Image Loaded Parameters Set Simulation Performed Output Displayed

3.3.1 Samples of Forms, Reports, and Interface

• Forms & Inputs: The GUI provides QLineEdit fields for numerical input and a QComboBox

for selecting preloaded camera types.

• Interactive Controls: Buttons for loading, saving, and clearing data, along with QDials for

adjusting roll, pitch, yaw, and tilt angles.

• Visualization: The PyQtGraph canvas displays the input and output images with an interactive

histogram.

18

4 IMPLEMENTATION

4.1 IMPLEMENTATION PLATFORM /ENVIRONMENT

The image simulation system is developed using Python and implemented on a Windows/Linux

environment. The key components of the implementation platform are:

• Programming Language: Python

• GUI Framework: PyQt5 (for creating an interactive graphical interface)

• Image Processing Library: GDAL (for handling GeoTIFF images)

• Visualization Library: PyQtGraph (for displaying images and histograms)

• Mathematical Computation: NumPy (for handling array-based image transformations)

• Development Environment: Spyder

• Hardware Requirements: A system with a minimum of 8GB RAM, multi-core CPU, and

GPU acceleration

The system runs efficiently on standard computing hardware, with optimizations to handle large

GeoTIFF files.

4.2 PROCESS / PROGRAM / T E CH N O L OG Y / MODULES

SPECIFICATION(S)

4.2.1 Process Flow

The image simulation follows these steps:

1. Loading the Input Image:

• Users load a GeoTIFF file using a file selection dialog.

19

CHAPTER 4. IMPLEMENTATION

• The image data is read using the GDAL library and processed into an array for simulation.

2. Setting Camera Parameters:

• Users manually enter or select a predefined camera type from a QComboBox.

• Parameters such as detector size, focal length, field of view, and altitude are applied.

3. Marking Positions (Single or Multi-Point Mode):

• Users click on the image or enter latitude-longitude coordinates to set markers.

• The system converts these to pixel coordinates for processing.

4. Footprint Visualization:

• A footprint button calculates and overlays the expected simulated area on the original

image.

5. Simulating the Image:

• The system applies geometric transformations based on camera parameters.

• The modified image is displayed on the PyQtGraph canvas.

6. Saving /Resetting Data:

• Users can save results, reset inputs, or clear markers as needed.

4.2.2 Technologies & Modules Used

The following technologies and modules are used in the implementation:

• PyQt5: GUI framework for interactive controls.

• PyQtGraph: Efficient image rendering and histogram display.

• GDAL: Reads and processes GeoTIFF images.

• NumPy: Handles mathematical transformations for image simulation.

4.2.3 Findings /Results /Outcomes

The implementation of the system resulted in the following outcomes:

• Accurate Image Simulation: The system successfully simulates how a camera would capture

an image in an unknown environment.

20

CHAPTER 4. IMPLEMENTATION

• Parameter Adjustment: Users can modify camera parameters and observe changes

dynamically.

• Footprint Visualization: The footprint feature helps users understand how the final image

will appear before simulation.

• Multi-Mode Simulation: Both single-point and multi-point simulation modes function as

expected.

• Efficient Processing: The use of NumPy and GDAL ensures that large GeoTIFF images are

processed efficiently.

4.2.4 Result Analysis /Comparison /Deliberations

The system was tested with various input images and camera parameters to evaluate its performance.

The key observations include:

• Comparison with Theoretical Expectations: The simulated images align with expected

outputs based on camera equations.

• Performance Metrics: Processing time varies based on image resolution and selected

parameters. Higher-resolution images take longer to process.

• User Interaction: The GUI provides an intuitive interface for parameter adjustments, making

the system user-friendly.

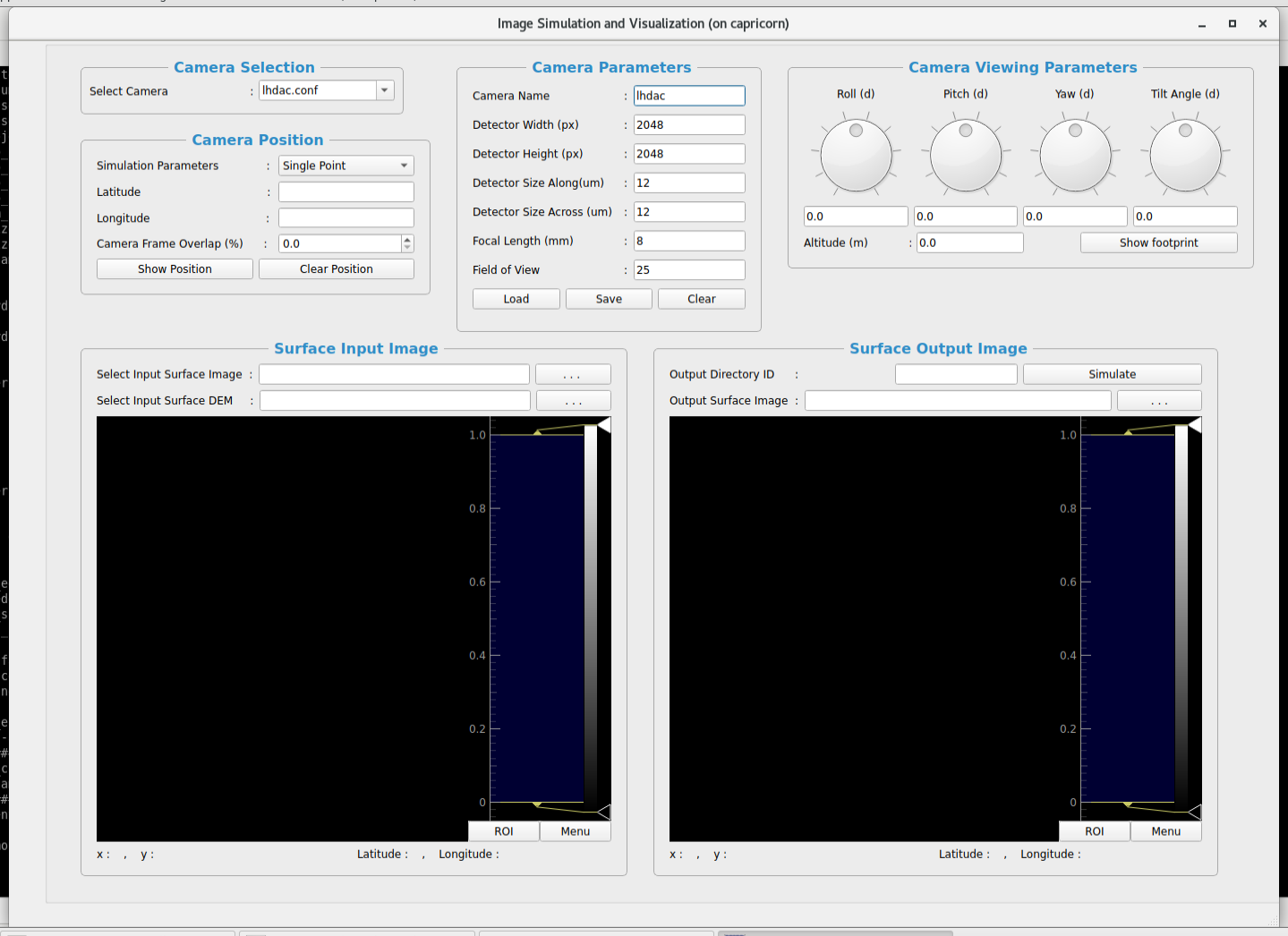
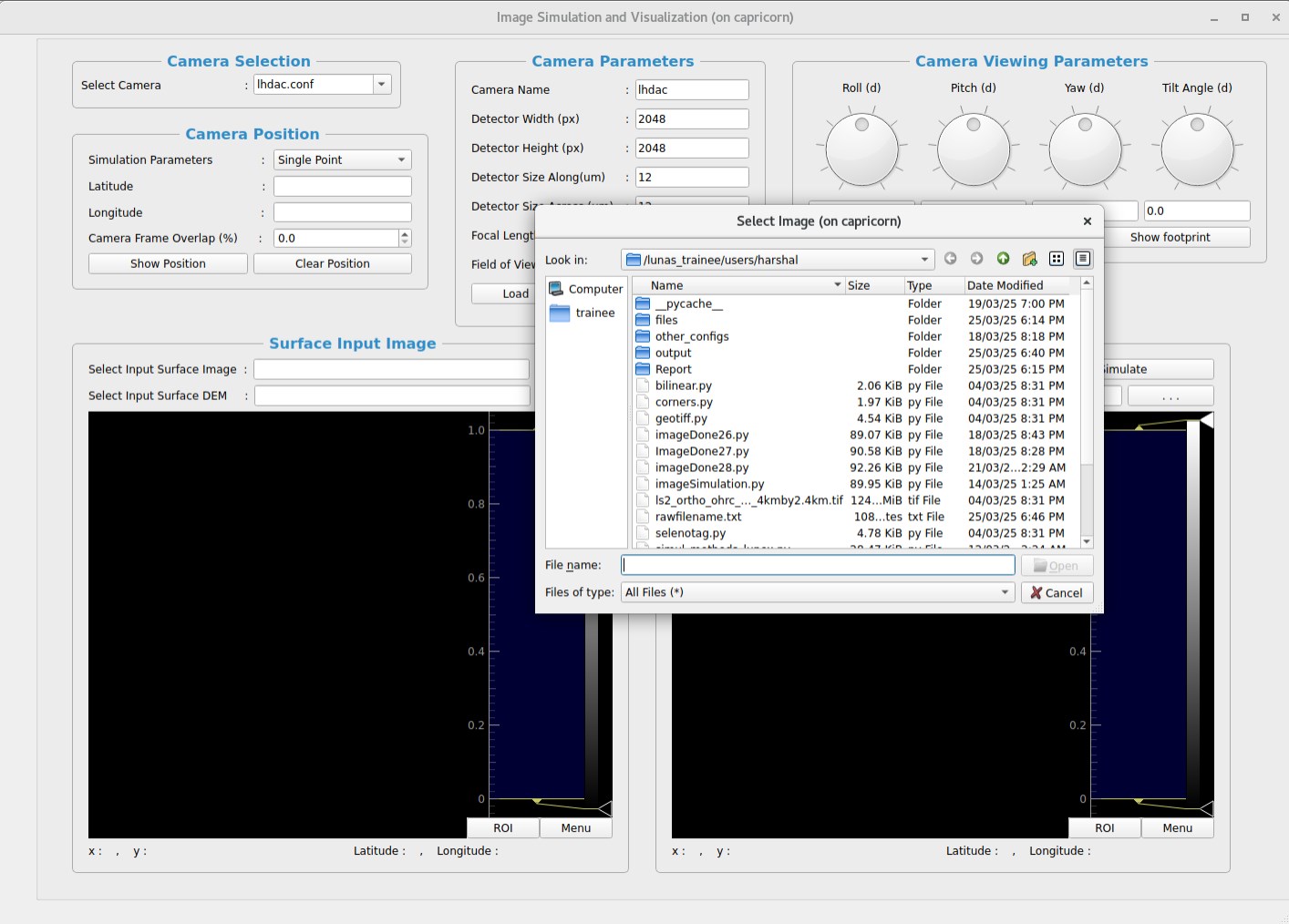
• Challenges: Some minor challenges were faced in marker precision, which were mitigated

using coordinate transformations.

4.2.5 Implementation :

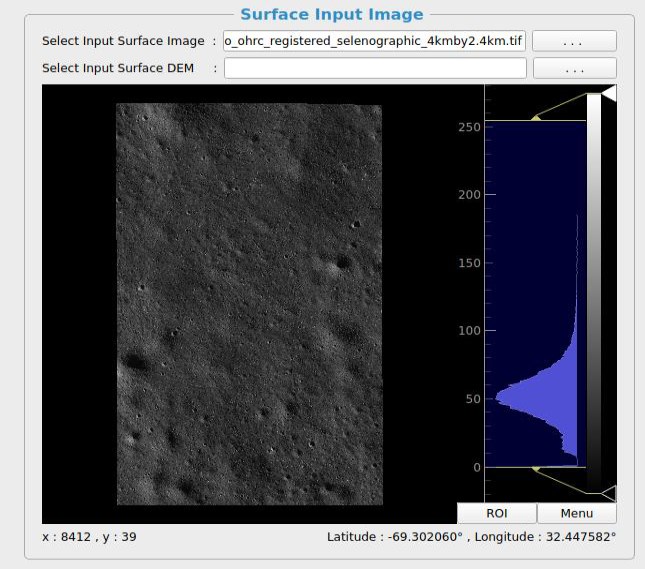
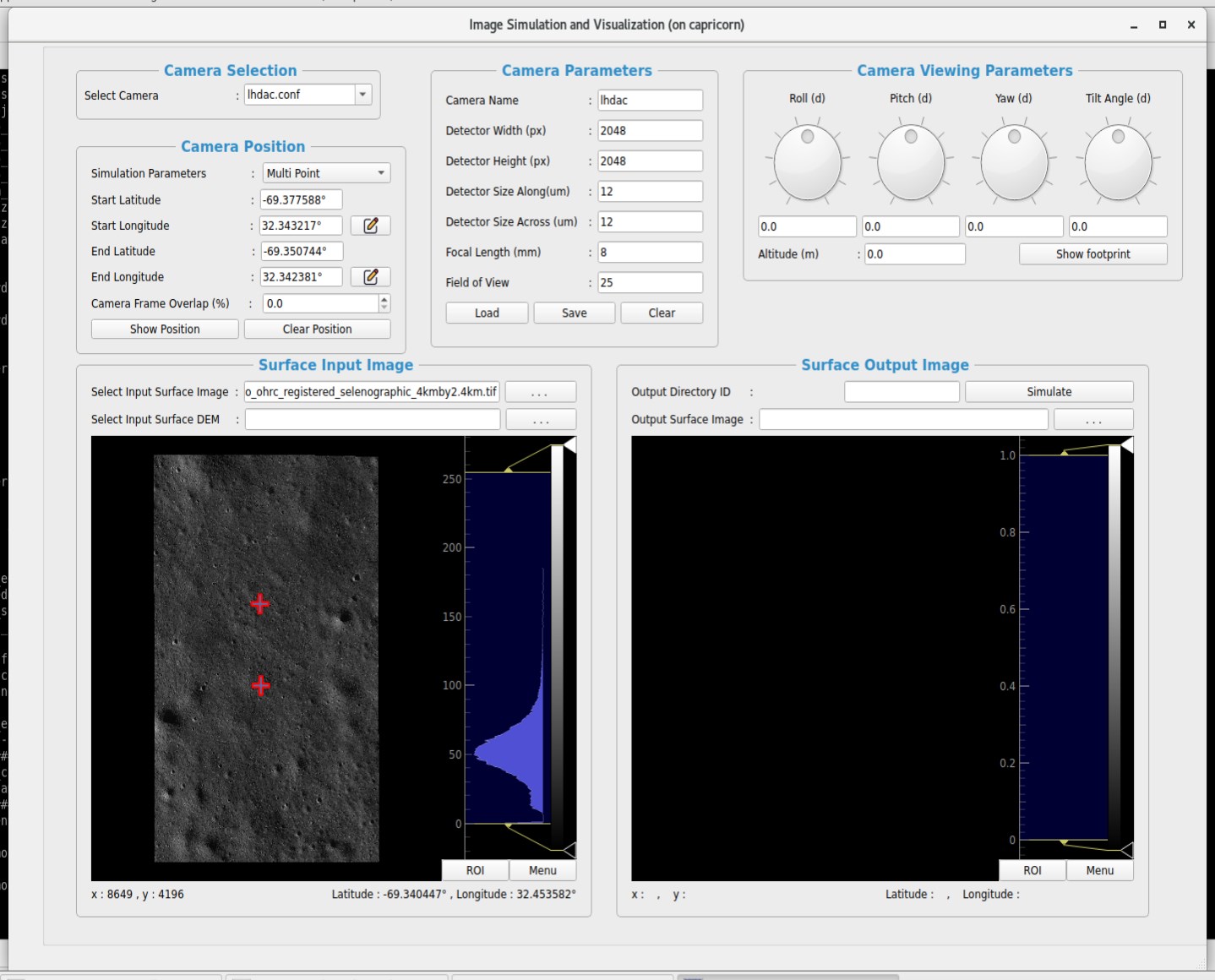
• Screenshots :

21

CHAPTER 6. IMPLEMENTATION

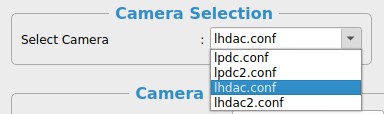
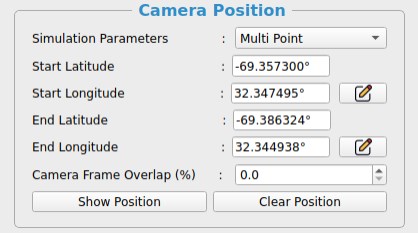
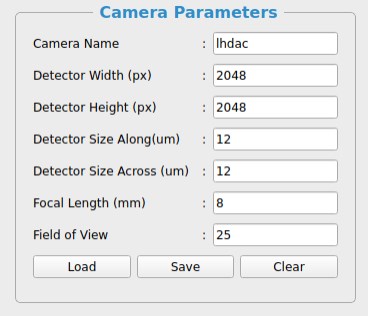
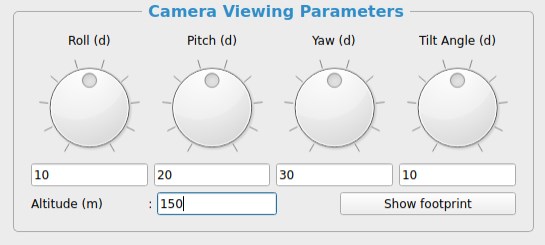
**Figure 4.1 Launching UI**

**Figure 4.2 Loading a GeoTif Image**



**Figure 4.3 Surface Input image**

**Figure 4.4 Visualization of Multipoint**

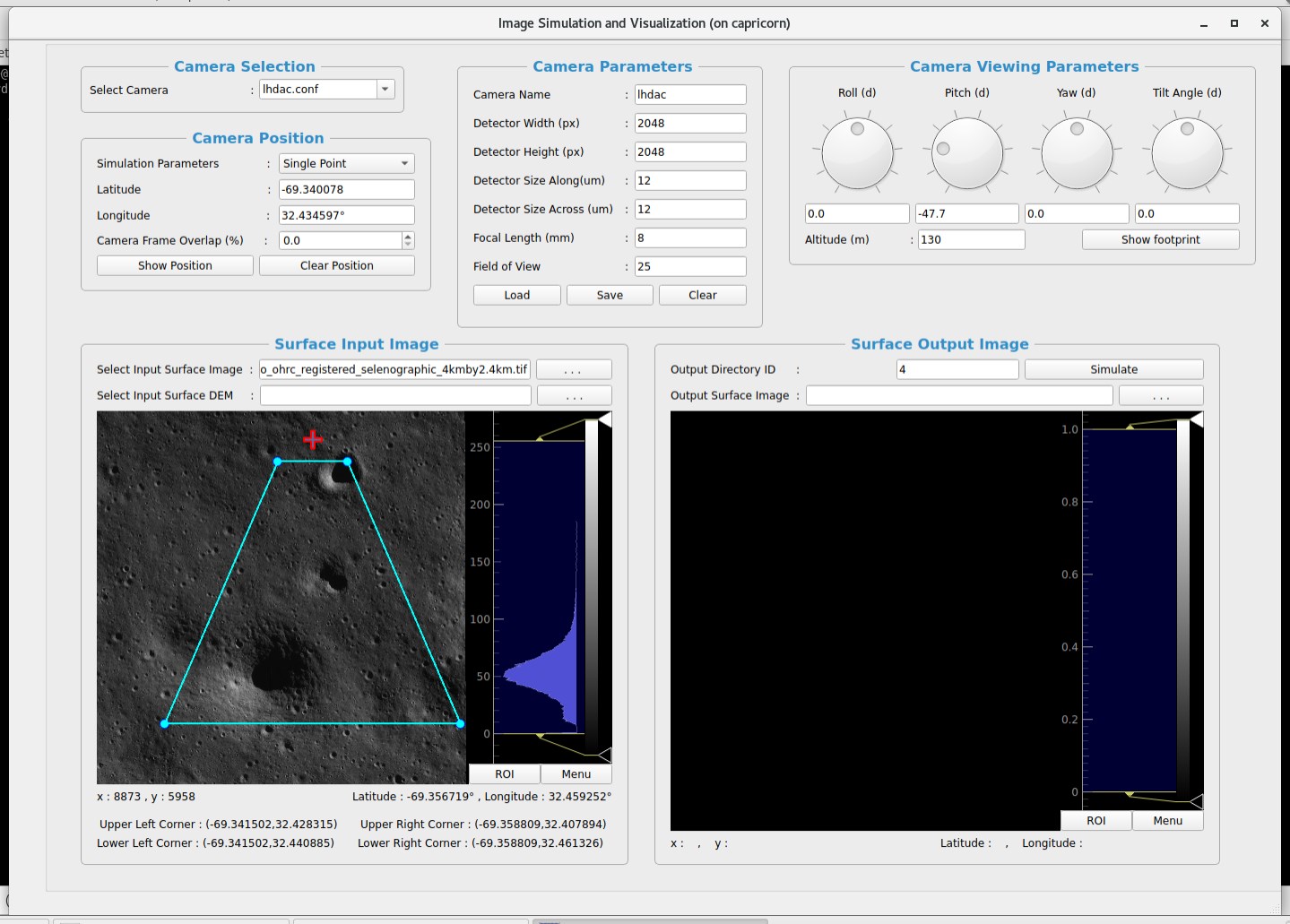
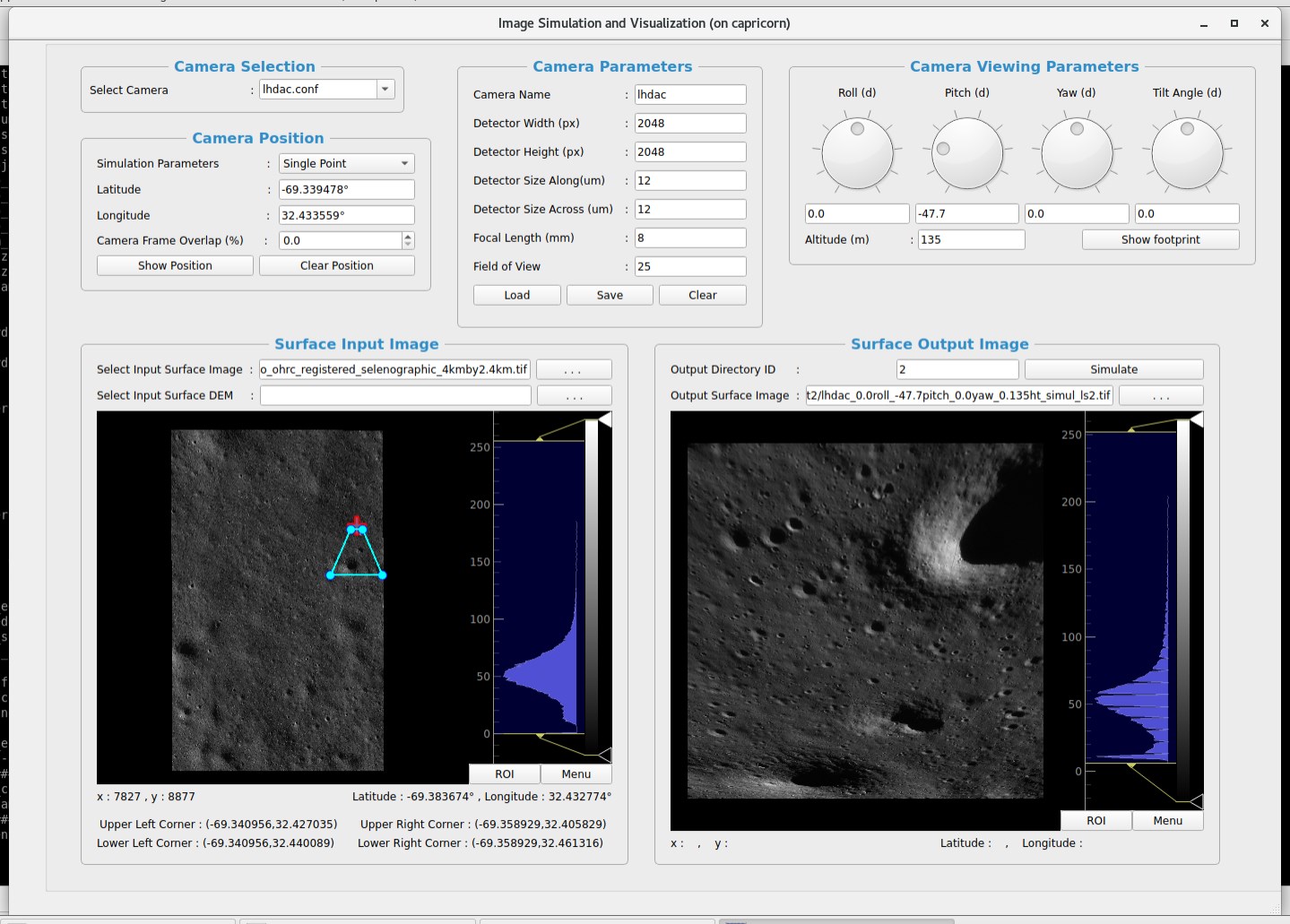


**Figure 4.5 Camera Selection**

**Figure 4.6 Multipoint Camera Selection**

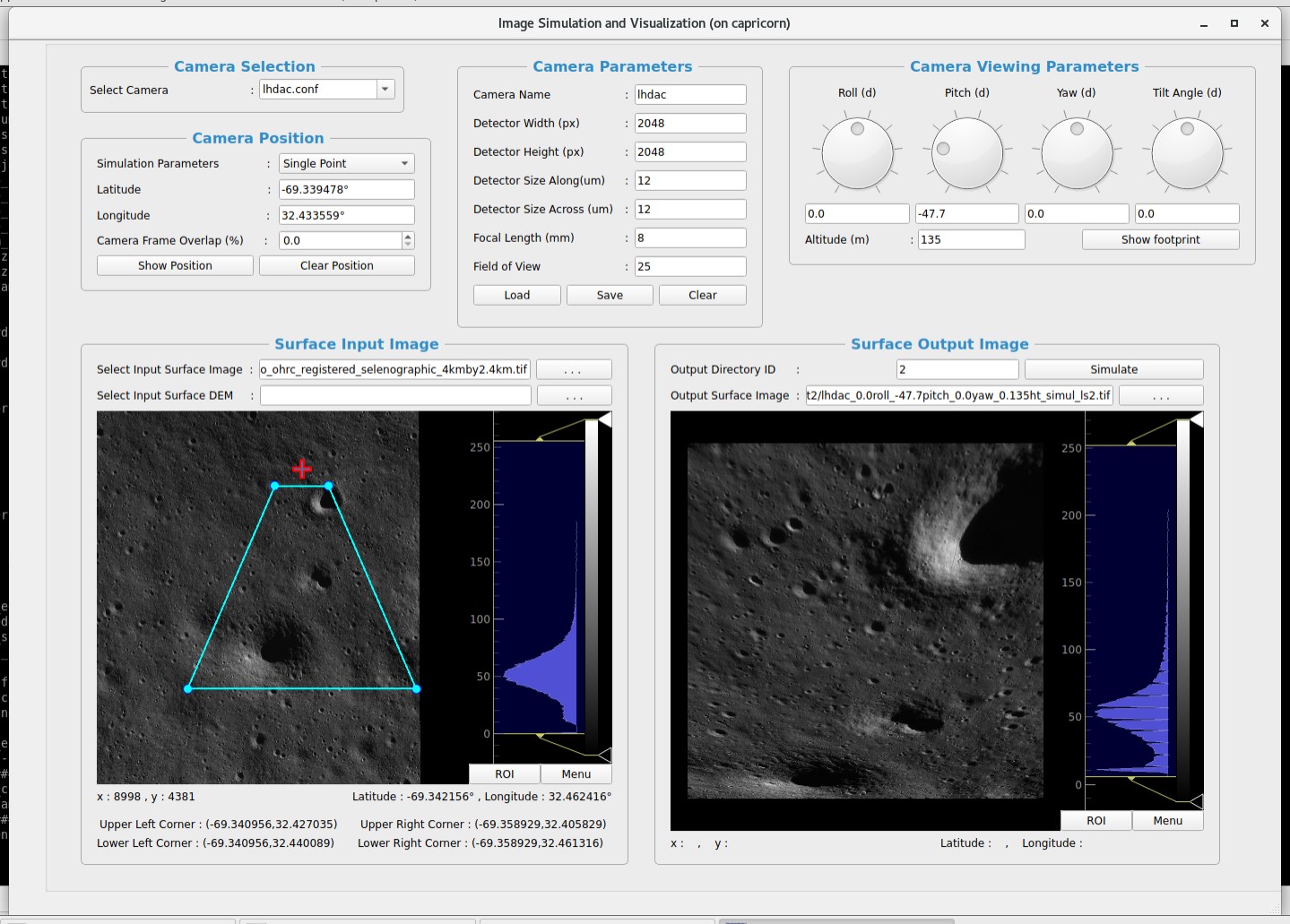
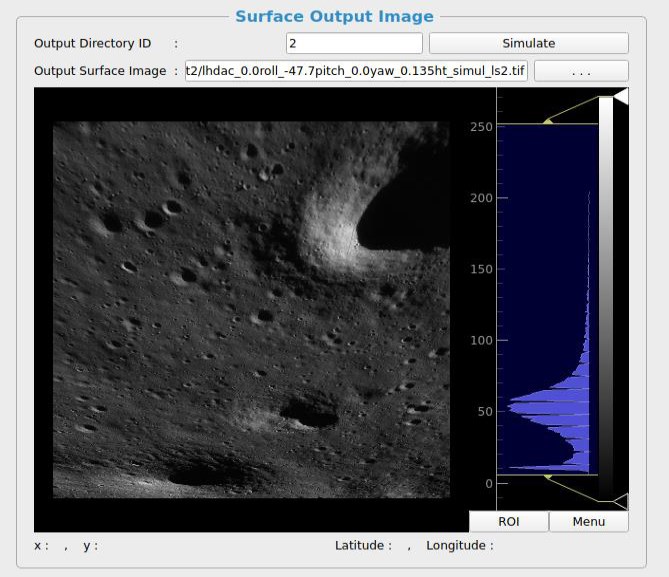
**Figure 4.7 Camera Parameters**

**Figure 4.8 Camera Viewing Parameters**



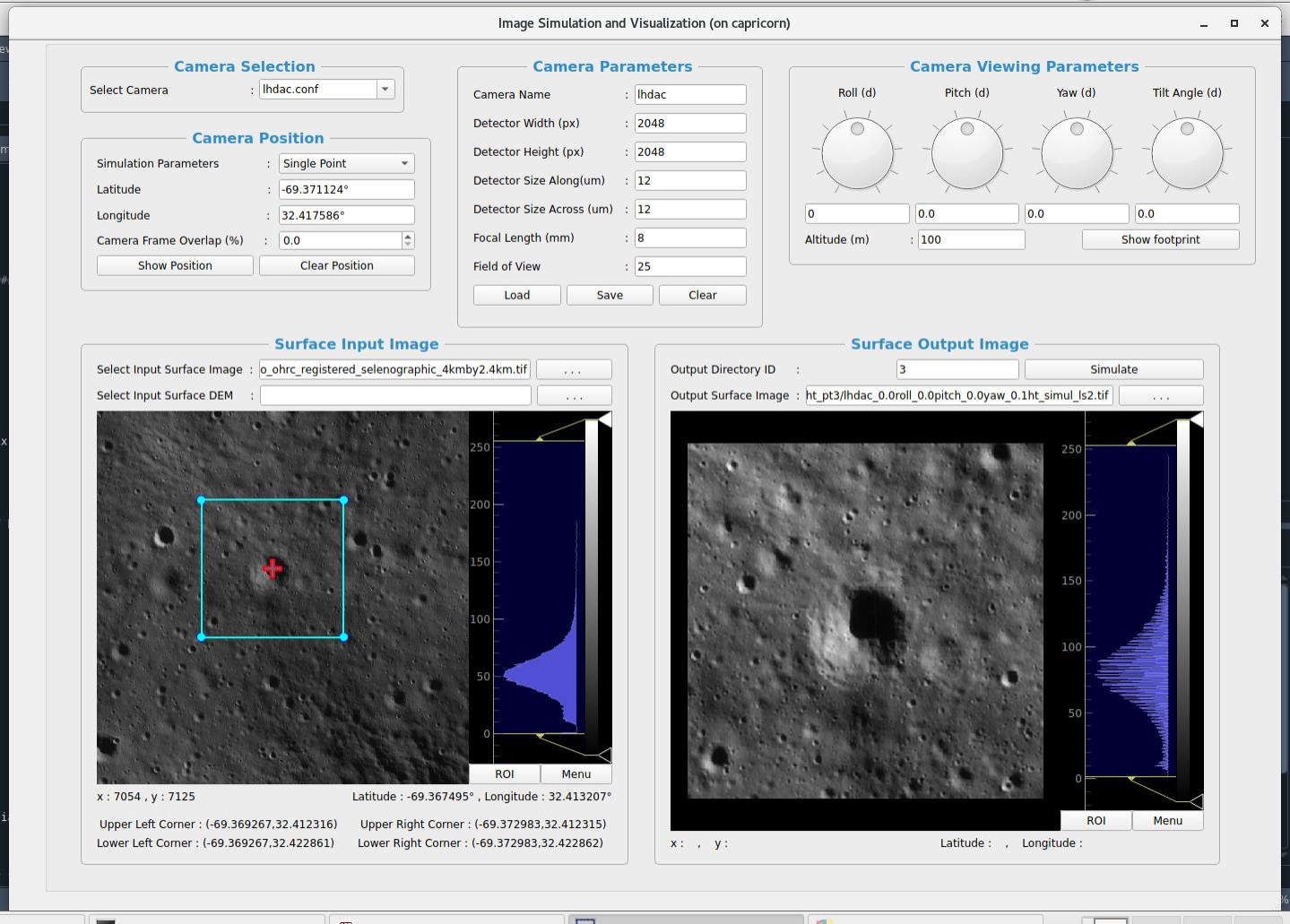
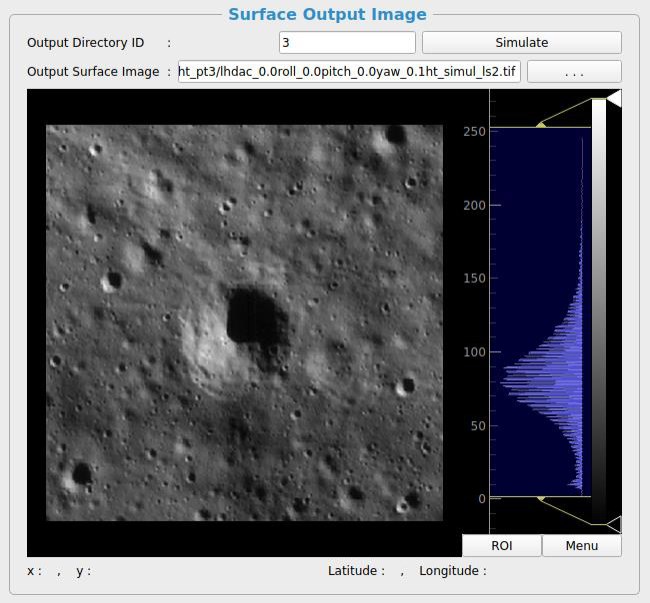
**Figure 4.9 Computing Footprint**

**Figure 4.10 Zoom out input image**



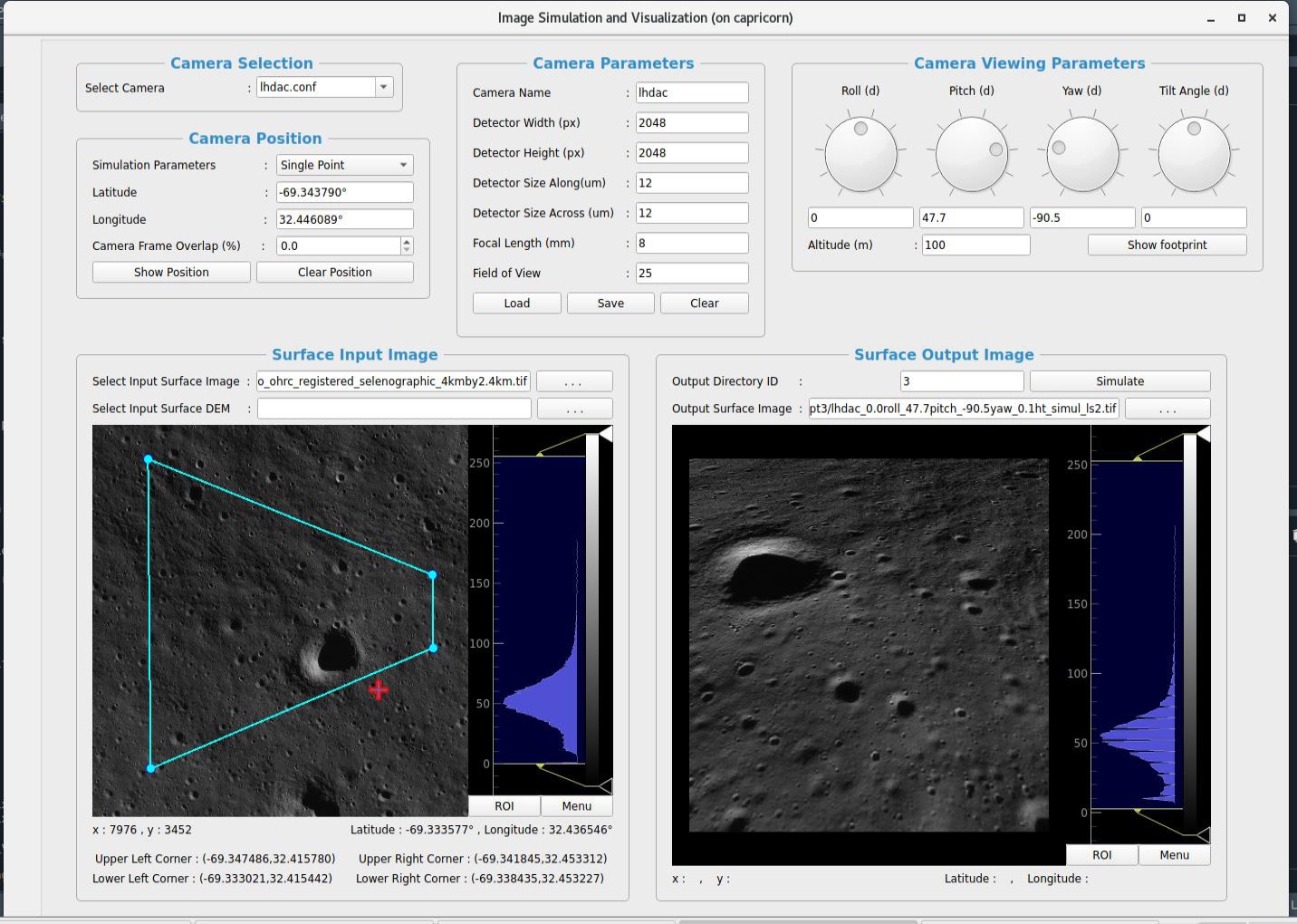
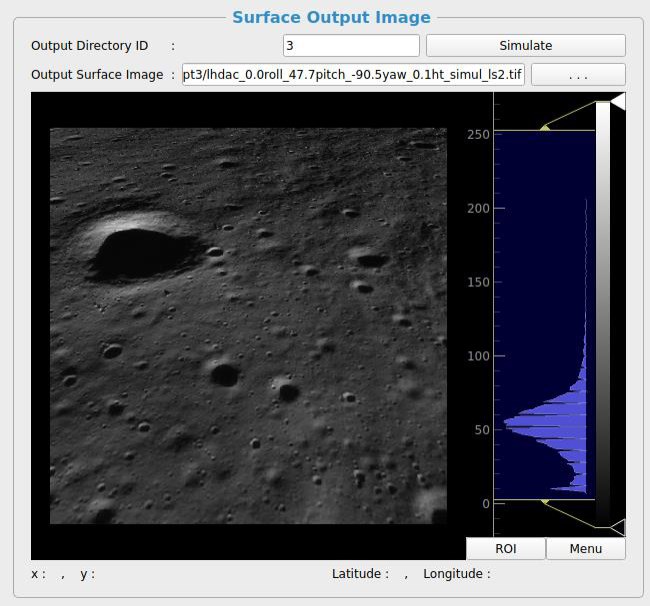
**Figure 4.11 Final Output**

**Figure 4.12 Output Image**



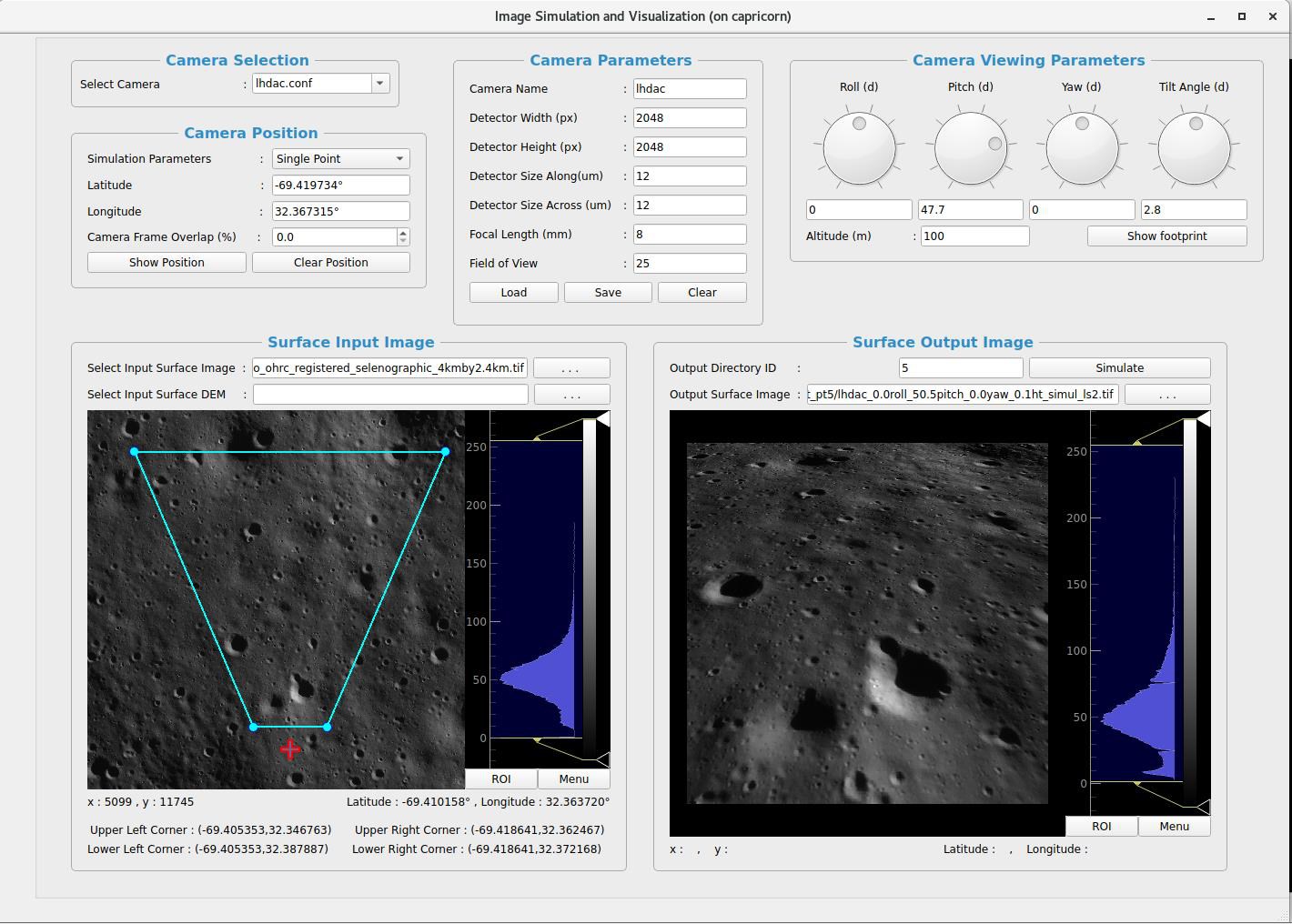
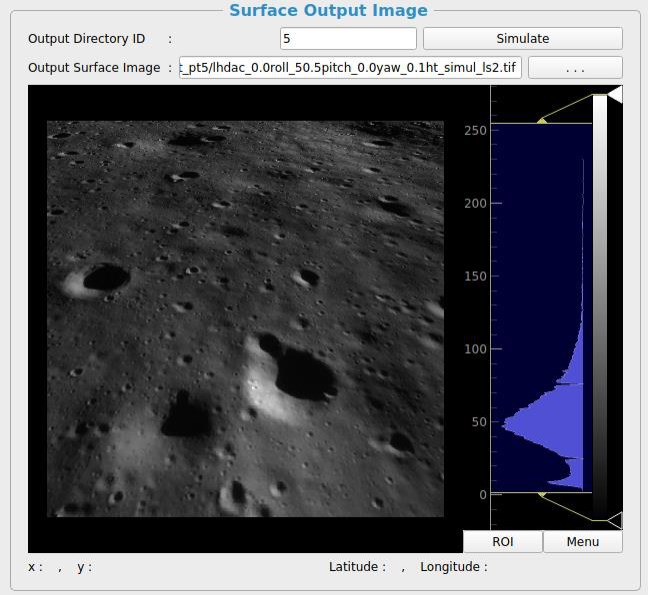
**Figure 6.13 Example\_1**

**Figure 4.14 Example\_1 Output**



**Figure 4.15 Example\_2**

**Figure 4.16 Example\_2 Output**



**Figure 4.17 Example\_3**

**Figure 4.18 Example\_3 Output**

5 TESTING

5.1 TESTING PLAN /S T R AT E G Y

The testing phase ensures that the image simulation system functions correctly and meets the

intended objectives. The strategy involves:

5.1.1 Testing Types Used

• Unit Testing – Individual components (e.g., image loading, parameter inputs, footprint

preview) are tested separately.

• Integration Testing – Ensures that different modules (e.g., GUI, parameter selection,

simulation engine) work together correctly.

• Functional Testing – Verifies that all features (e.g., single/multi-point simulation, marker

placement) perform as expected.

• Performance Testing – Checks system efficiency with different image sizes and parameter

variations.

• User Acceptance Testing (UAT) – Ensures the system meets user requirements for camera

testing and design.

5.1.2 Testing Environment

• Platform: Linux

• Software Dependencies: Python, PyQt5, PyQtGraph, GDAL, NumPy

• Hardware: System with at least 8GB RAM and multi-core processor

5.2 TEST RESULTS AND ANALYSIS

Multiple test cases were executed to verify the system’s performance and accuracy. The following

table summarizes key test cases:

30

CHAPTER 5. TESTING

5.2.1 Test Cases

Test ID Test Condition Expected Output Actual Output Remarks

TC-01 Load GeoTIFF image Image loads Image loads correctly Pass successfully

TC-02 Enter valid camera parameters

Parameters accepted and applied

Parameters stored and Pass used

TC-03 Enter invalid camera parameters

Error displayed

message Error handled Pass correctly

TC-04 Single-point marker placement

Marker appears at correct lat-long

Marker placed Pass accurately

TC-05 Multi-point placement

marker Two markers placed correctly

Markers appear as Pass expected

TC-06 Footprint preview Correct displayed

footprint Footprint aligns with Pass expected region

TC-07 Run image simulation Simulated image is generated

Image output is Pass accurate

TC-08 Save simulated image

TC-09 Clear parameters

Image saved directory

All fields reset

in File successfully

Fields properly

saved Pass

cleared Pass

TC-10 Performance with high-res image

Processing within reasonable time

Acceptable Pass (minor delay) performance

Table 7.1: Test Cases for Image Simulation System

5.3 ANALYSIS OF TE ST RESULTS

• Functionality: All core features performed as expected, with accurate image simulation.

• Usability: The GUI is intuitive, with smooth user interactions.

• Performance: The system handles various image resolutions well, but higher-resolution

images slightly increase processing time.

• Error Handling: Proper validation prevents invalid inputs, improving system robustness.

31

6 CONCLUSION AND SUMMARY

6.1 CONCLUSION AND SUMMARY

The image simulation system was developed to evaluate how a camera would capture images in an

unknown environment. By incorporating camera-specific parameters such as detector height, width,

field of view (FOV), focal length, and altitude, the system provides an effective way to analyze and

design cameras before actual deployment.

The system meets its objectives by providing a useful tool for Image simulation and being

helpful for studying images.

6.2 LIMITATIONS

• Processing Time for High-Resolution Images: When working with very large GeoTIFF

images, processing can take longer than expected. Optimizations in computation can help

reduce delays.

• Limited Camera Parameter Customization: The system currently supports predefined

parameters and manual input. Future versions could allow users to import real-world camera

calibration data.

• No Real-Time 3D Visualization: The system operates in a 2D simulation environment.

Adding a 3D rendering option could improve visualization and understanding of camera

perspectives.

• Georeferencing Accuracy: While the system accurately places markers based on latitude-

longitude coordinates, minor discrepancies may arise due to projection transformations.

• No External Data Integration: Currently, the system processes only user-provided images.

Future enhancements could support integration with external satellite data sources.

32

6.3 FUTURE WORK

• Marker History & Undo Feature: When working with very large GeoTIFF

Let users Undo/Redo marker placements or toggle through marker history, making experimentation easier without losing previous selections.

• Configuration Templates: Allow users to save simulation setups as templates — these could include camera types, frequently used lat-long pairs, or entire parameter sets.

• Add Keyboard Shortcuts for Common Actions: Implement hotkeys (e.g., Ctrl+L for Load, Ctrl+S for Save, F for Footprint) to speed up workflows for power users.

• 3D Terrain Visualization: Incorporate 3D elevation models (DEMs) for realistic terrain simulation.

• No External Data Integration: Currently, the system processes only user-provided images.

Future enhancements could support integration with external satellite data sources.

33