

# Improving the Development Workflow of the SETSM Photogrammetry Software

Samuel Khuvis<sup>1</sup>, Judith Gardiner<sup>1</sup>, Myoung-Jong Noh<sup>2</sup>, Caleb Lehman<sup>1</sup>, Ian Howat<sup>2,3</sup>, Karen Tomko<sup>1</sup>

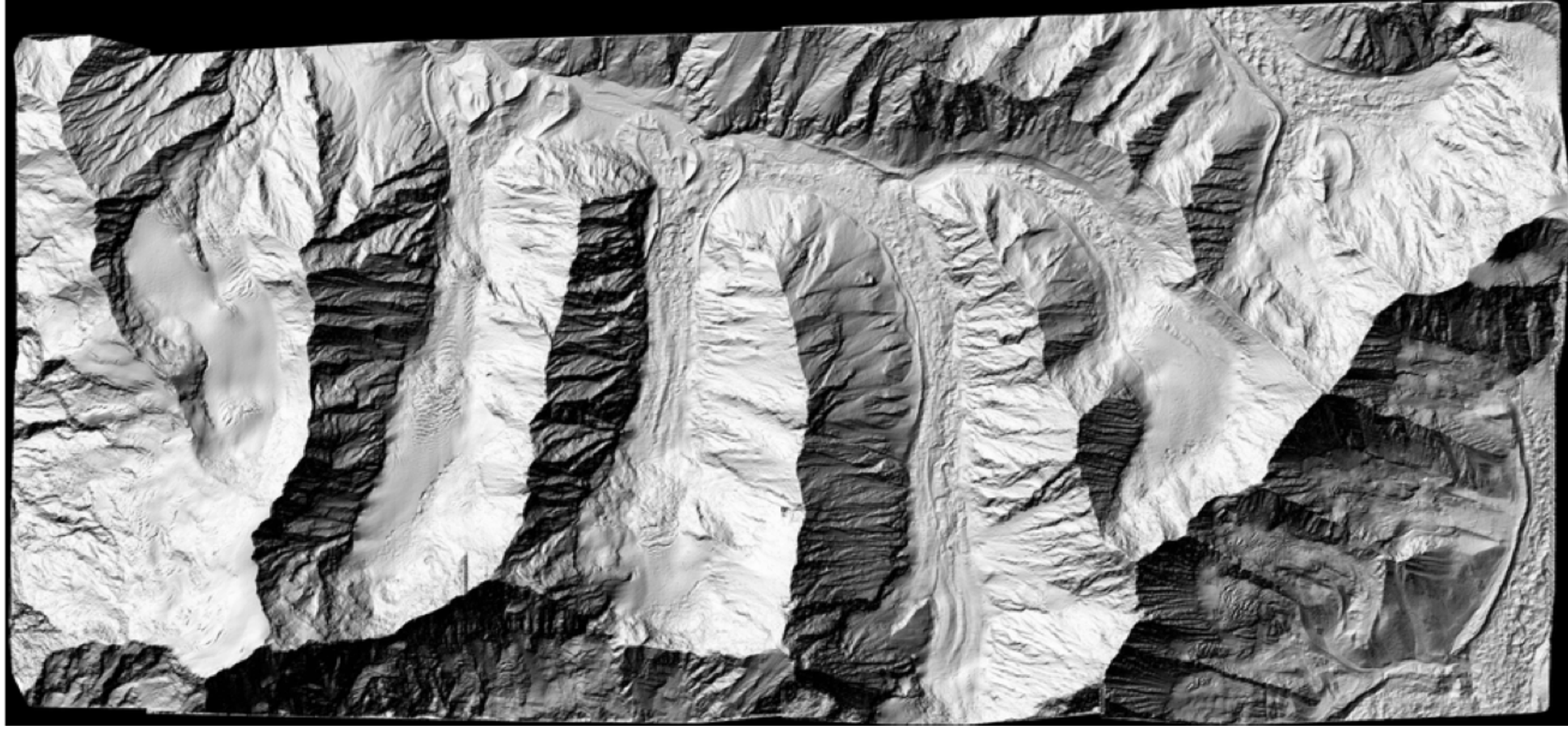
<sup>1</sup>Ohio Supercomputer Center

<sup>2</sup>Byrd Polar and Climate Research Center, The Ohio State University

<sup>3</sup>School of Earth Sciences, The Ohio State University

## Introduction

The Surface Extraction by TIN-based Search-space Minimization (SETSM) software is used to produce digital elevation maps (DEMs) from satellite imagery [1, 4]. The goal of SETSM is to automatically extract a stereo-photogrammetric DEM from pairs of images without any user-defined or a-priori information and using only the sensor Rational Polynomial Coefficients for geometric constraints. The software is written entirely in C/C++ and is only dependent on the libtiff and libgeotiff libraries. The SETSM algorithm constructs a Triangular Irregular Network (TIN) in object-space domain to minimize the necessary search space and it employs the coarse-to-fine and vertical line locus strategies.



Nepal DEM

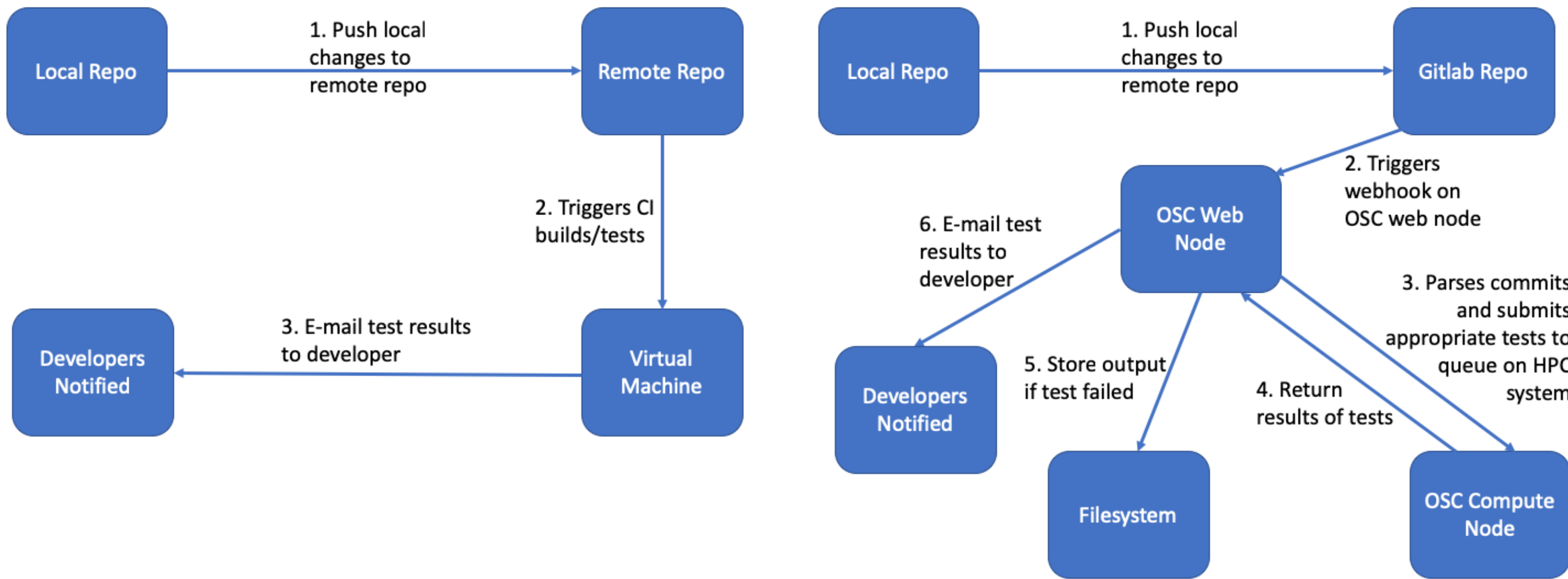
The challenges in development are:

- Concurrent optimization for performance [3] and development by domain scientists.
- Developers working on optimization do not have domain expertise and cannot easily determine whether a modification to the code that changes the output DEM is acceptable or not.
- SETSM has high numerical sensitivity so we would like to verify that it produces consistent results for the same inputs.

This poster presents a workflow to deal with these challenges.

## Test Workflow

A Continuous Integration-like workflow is used to verify that changes to the code do not have any unintended effects. Traditional CI consists of a number of unit tests that can be built and run on a virtual machine. Scientific codes cannot always be fully tested with unit tests. A SETSM run can take hours to run and multiple test cases are necessary to fully test for errors. So, we must queue jobs to an HPC system to run the full suite of tests. Also, input/output data sizes can be large, so we must carefully store results of runs.



Traditional CI workflow

OSC test workflow

The SETSM code is hosted in a Gitlab repository configured with webhook integration. When a commit or comment is pushed to the repository, a webhook posted at OSC [5] is run. This webhook launches a Python script that parses the message and launches appropriate tests. Tests are managed using the ReFrame framework from CSCS [2]. If the test fails, output is stored in a unique subdirectory of the project directory. An e-mail with the results of the tests and the path to output is sent to appropriate developers.

## Available Tests

- **Consistency Test:** run the same executable multiple times and verify that the output is the same each time.
- **Version Test:** run the code with different versions of a compiler and verify that the output is the same.
- **Reference Comparison Test:** compare the output of the code to a previous version of the code that has been verified as correct.

## Results

This test workflow has been able to identify errors introduced to the code. For example, by running the **Consistency Test**, which runs the same executable three times and compares the outputs, we were able to find that changes to the code were no longer producing consistent results. Two of the resulting image files were:



Reference DEM



New DEM

These tests are run after every major change to the code, so we know which version of the code still produced consistent results. By comparing the two versions of the code, we were able to find that this issue was caused by the addition of an OpenMP pragma to a loop with dependencies.

## Future Work

### Current limitations:

- If we expect DEMs to match exactly we can check that outputs match exactly, otherwise we must send DEM to domain scientist for verification.
- Long runtimes for benchmarks.
- Large output files.

### Planned Development:

- Check that DEM is within acceptable tolerance of a reference DEM.
- Determine appropriate tests in script instead of explicitly specifying them in commit/comment.
- Reduce test durations by splitting runs into separate submissions.
- Reduce size of stored outputs of tests.

## References

[1] <https://mjremotesensing.wordpress.com/setsm>.  
[2] <https://github.com/eth-cscs/reframe>.  
[3] J. D. Gardiner, K. A. Tomko, M.-J. Noh, and I. M. Howat. Code optimization and stabilization for a high-resolution terrain generation application. In *Proceedings of the Practice and Experience on Advanced Research Computing*, PEARC '18, pages 82:1–82:3, New York, NY, USA, 2018. ACM.  
[4] M.-J. Noh and I. M. Howat. The surface extraction from tin based search-space minimization (setsm) algorithm. *ISPRS Journal of Photogrammetry and Remote Sensing*, 129:55 – 76, 2017.  
[5] Ohio Supercomputer Center. Ohio Supercomputer Center. <http://osc.edu/ark:/19495/f5s1ph73>, 1987.

## Acknowledgments

This research is supported in part by National Science Foundation awards 1559691 and 1543501. DEMs produced using data from DigitalGlobe, Inc.

