Hochschule für Technik Stuttgart
University of Applied Sciences

Fakultät Vermessung, Informatik und Mathematik

Photogrammetry and Geoinformatics

Prof. Dr.-Ing. E. Gülch

Advanced Topics in Photogrammetry and Remote Sensing

Structure from motion (SFM)

SS2024

Gökhan Yücesan

20015385

Mbanzi Valere Iradukunda

20015257

TASK A: Getting Started with Visual Sfm

Visual Sfm QT-based visualization software which can be used to display 3D point clouds from .out and .ply files, which are the outputs of bundler_sfm. It was developed by Changchang Wu and distributed freely for personal, non-profit, and academic use. It consists of a GUI application for 3D reconstruction using structure from Motion (SFM). In VisualSFM, the "Compute Missing Matches" function improves 3D reconstruction by finding connections between image features missed initially. The "Compute 3D Reconstruction" function generates the initial structure from motion. Lastly, the "Create Dense Reconstruction" function adds detail to the 3D model for a more accurate representation. These functions are essential for creating comprehensive and detailed 3D reconstructions.

TASK B. Application of the software

For the application of VisualSFM software, we captured images of two objects using a single mobile phone. The specifications of the phone used are detailed in Table 1 below.

Specificaions	
Model	Iphone 10
Camera System	Dual 12-megapixel camera system
Lenses	Wide-angle and telephoto lenses
Wide-angle Lens Aperture	f/1.8
Telephoto lens Aperture	f/2.4
Focal length	35mm
ISO	ISO-20 to ISO-5000
Dimensions	4032 pixels x 3024 pixels
Average size	3-4 Megabytes

Table 1: Camera Phone Specifications

Processing Step A

Recording of images.

During the image capture session using a smartphone, we focused on capturing two distinct objects. The first object was a monument situated in Mittlerer Schlossgarten, while the second was a war scene located inside one of the buildings on the HFT Campus, as seen in Figure 1. Throughout the capture process, we made a deliberate effort to photograph the objects from multiple angles, ensuring comprehensive coverage and facilitating the subsequent 3D reconstruction process. For object 1 we captured 132 images, while for second object it was 68.

Figure 1: Object 1 and 2

Orientation of images using the software VisualSFM

For this stage we first loaded our multiple images and then we computed the missing matches. For the first object the process took 64 minutes.

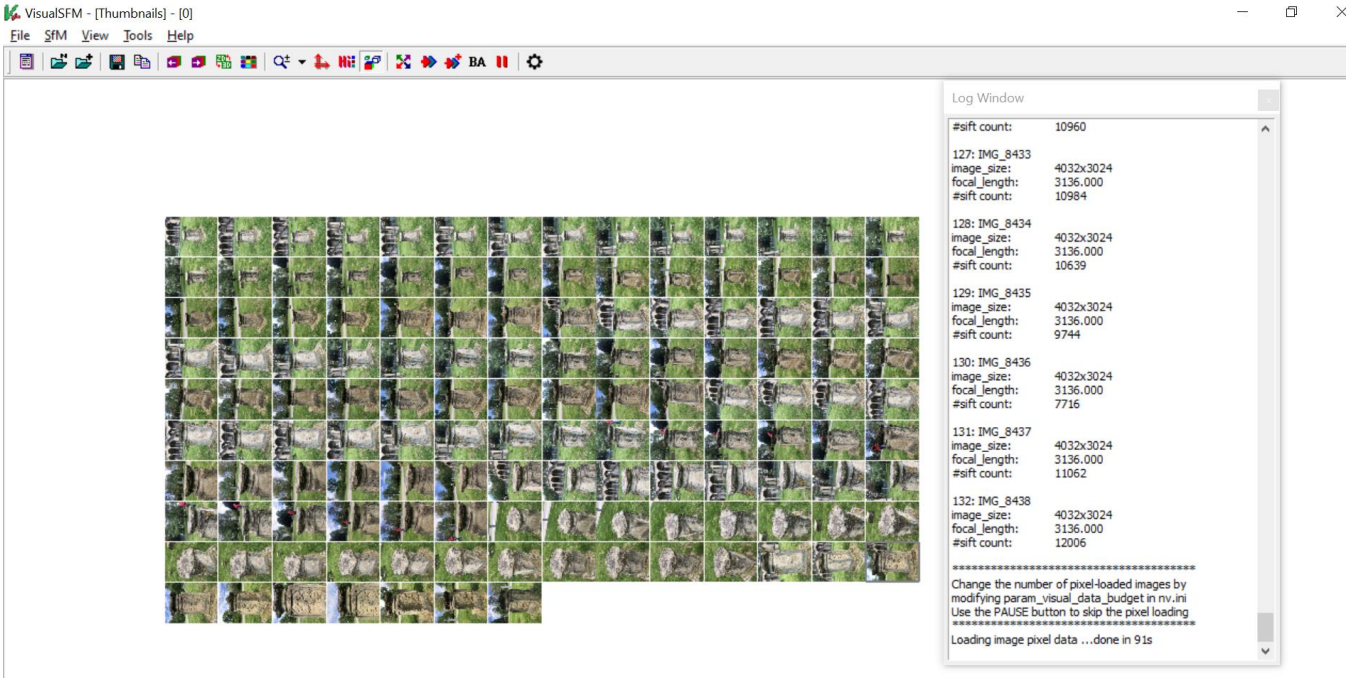

Figure 2: Orientation of object 1

For second objects the process of computing missing matches took 31.2 minutes.

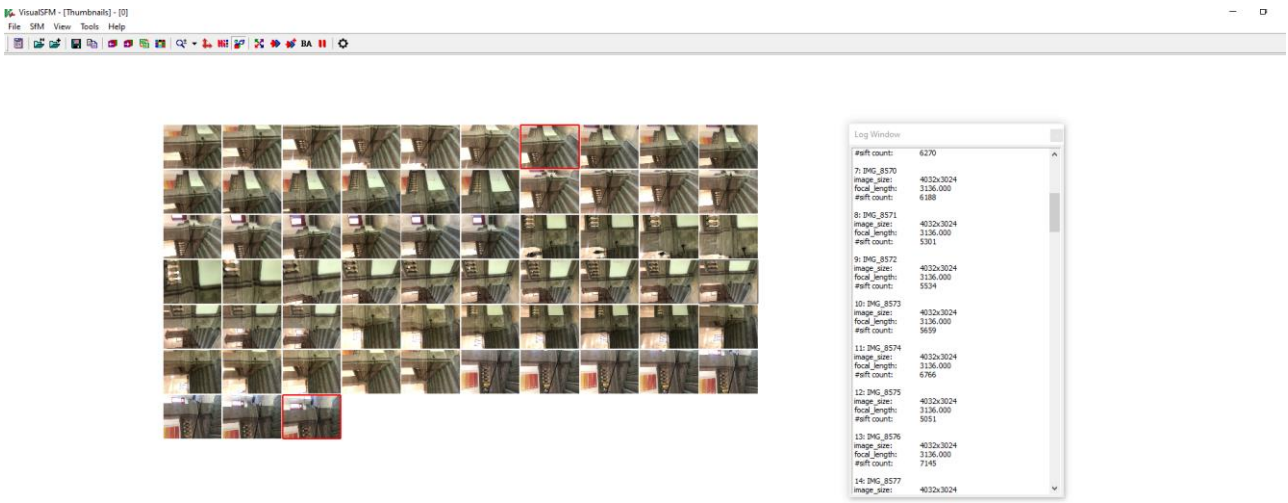

Figure 3: Orientation of object 2

Generation of a sparse point cloud with the software VisualSFM

To generate a sparse point cloud, we used a tool called 'Compute 3D Reconstruction' which reconstruct the scene's 3D geometry by estimating camera positions and sparse 3D points, creating an initial sparse point cloud model.

Figure 4: Sparse points for object 1

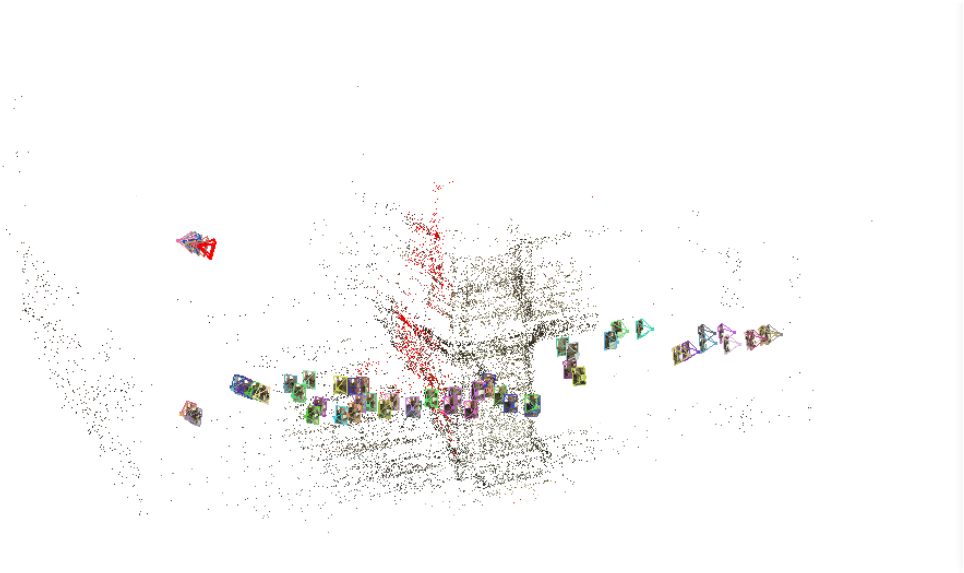

Figure 5: Sparse points object 2

Analysis

The 3D reconstruct function analyzes the input images to determine the spatial relationship between them and reconstruct the 3D geometry of the scene. It employs structure-from-motion (SfM) algorithms to estimate the camera poses (positions and orientations) relative to each other and the sparse 3D points representing the scene. By triangulating the matching features across multiple images, VisualSFM calculates the 3D coordinates of these points and generates an initial sparse point cloud model of the scene. From the two objects, we could see that they was a big difference on how the points are distributed as for object 1 the points was well distributed compared to object 2. We noticed that some of the reasons might have been the different of captured images between the two objects, their location as the second object was indoor and the types of textures that the two objects were made of.

Processing steps B

Generation of dense point cloud with CMVS/PMVS2

CMVS (Clustering Views for Multi-view Stereo) and PMVS2 (Patch-based Multi-view Stereo) are a pair of software tools used for dense 3D reconstruction from images. The tool can be found inside the Visual SFM SW and the results for both objects can be seen in Figure 6 and **Error! Reference source not found.**.

Figure 6: Dense point cloud of object 1 generated by CMVS/PMVS2

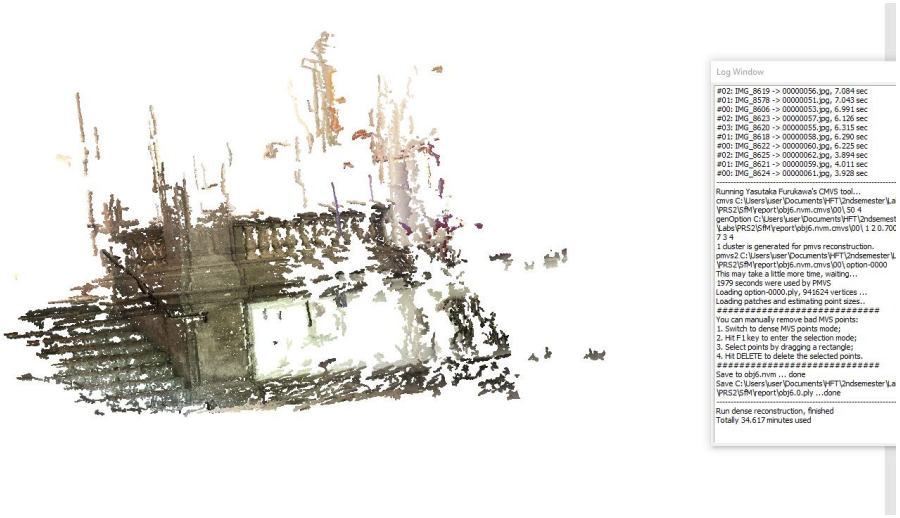

Figure 7: Dense point cloud of object 2 generated by CMVS/PMVS2

Generation of dense point cloud and texturing with CMPMVS

We run CMPMVS software using the command prompt as seen in Figure 8. For Object 2 it took 2hours and 47 munites while for object 1 it took 4 hours 30 munites. As seen in figures below, we got different outputs as the results from the CMPMVS, in Figure 11 we can see the results from the video that we received as results where we could see the textured and shaded images together with the original images for both objects. Another result we got was the 3D dense cloud points from both objects that was visualized in Meshlab as seen in both Figure 9 and Figure 10.

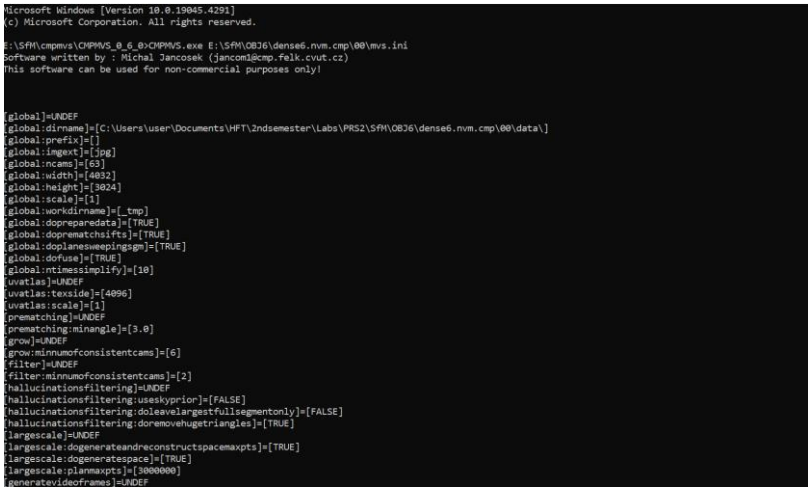

Figure 8: CMPMVS command prompt

Results from CMPMVS

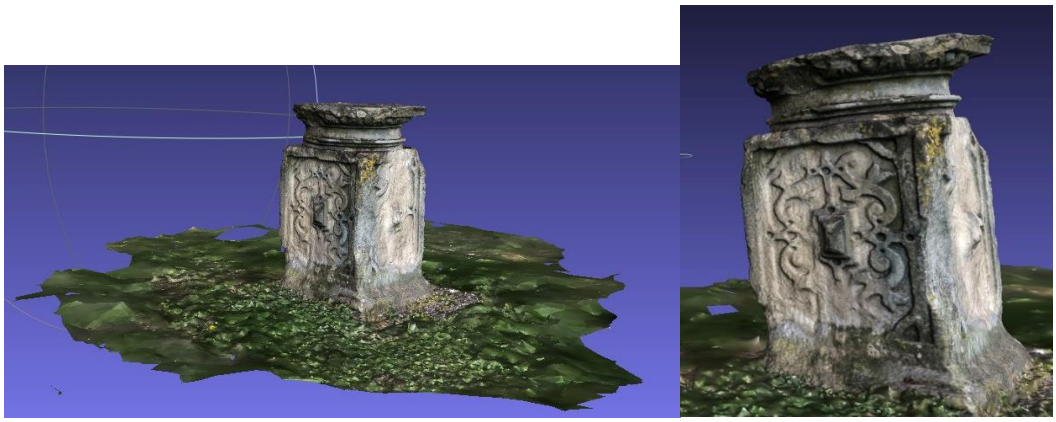

Figure 9: 3D dense cloud points of Object 1 after CMPMVS

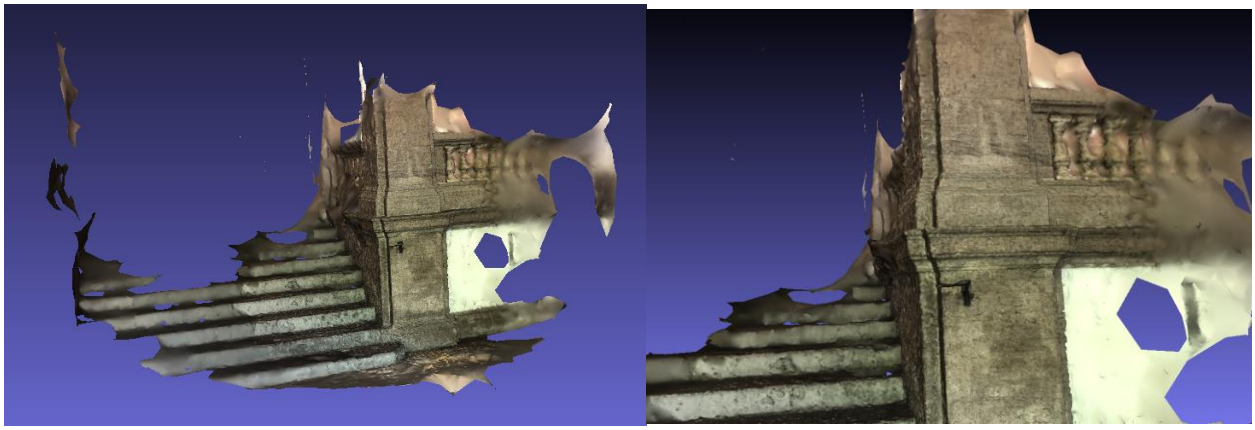

Figure 10: 3D dense cloud points of Object 2 after CMPMVS

Figure 11: Results from videos

Comparison between CMVS/PMVS2 and CMPMVS

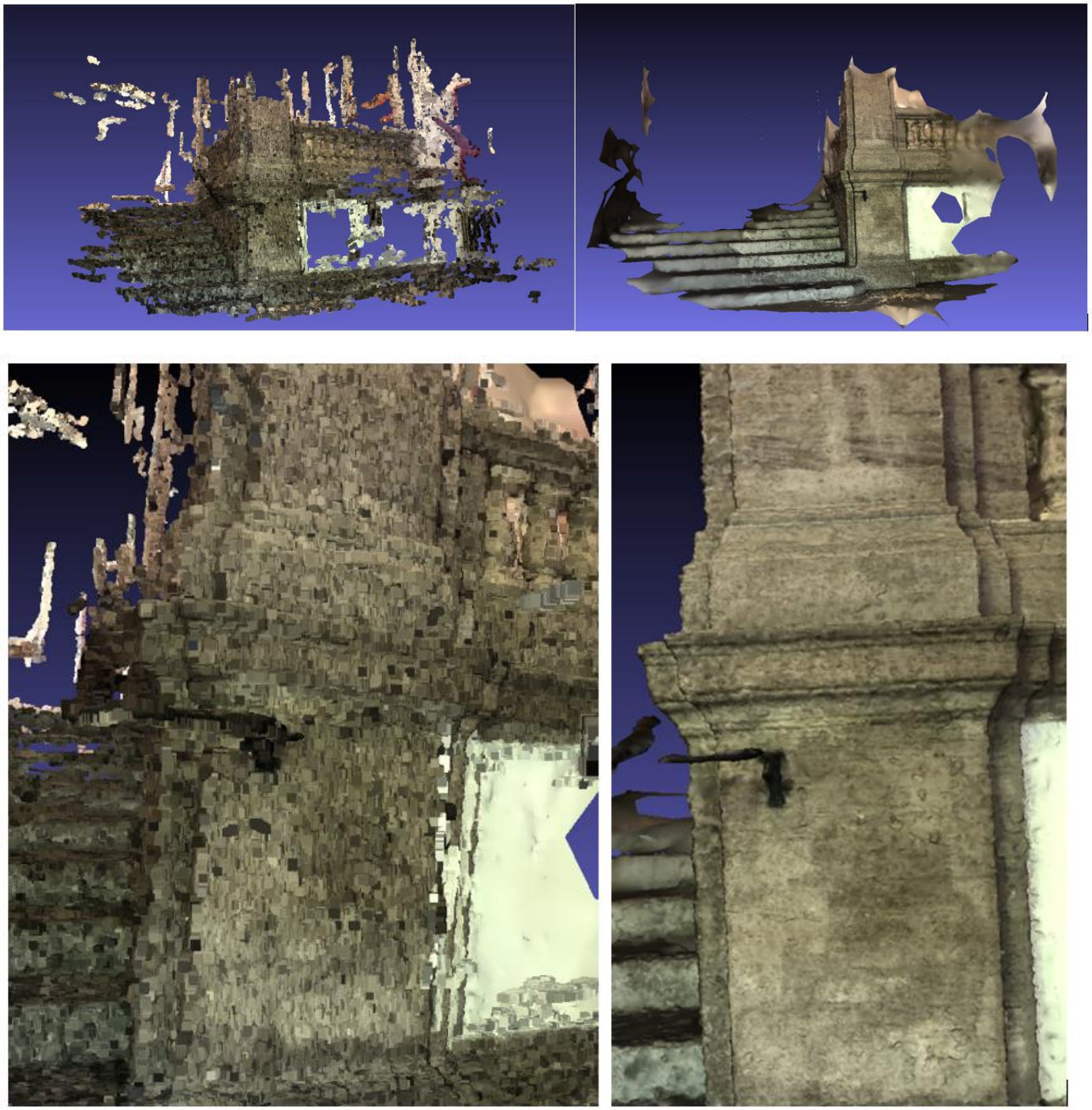

Figure 12: 3D dense cloud CMVS/PMVS2 (left) and 3D dense cloud CMPMVS (right) for object 2

Figure 13:3D dense cloud CMVS/PMVS2 (left) and 3D dense cloud CMPMVS (right) for object 1

Analysis

As seen in both Figure 12 and Figure 13, we can easily see that results from CMPMVS are better than results from CMVS/PMVS2. Textures in CMPMVS are well defined compared to CMVS/PMVS2. Although the results of the CMVS/PMVS2 seem to not be as good as the results of CMPMVS, Different techniques from meshlab software were used to improve their textures as seen in the last part of the report.

Optional: MESHLAB

For meshlab, we started by deleting unnecessary points of our objects. The next step was to generate the polygonal surface using Screened Poisson surface reconstruction, which creates watertight surfaces from oriented point sets. And for final step we generated texture using the Parameterization + texturing from registered rasters which creates some patches that correspond to projection of portions of surfaces onto the set of registered rasters.

Figure 14:Object 1 in meshlab

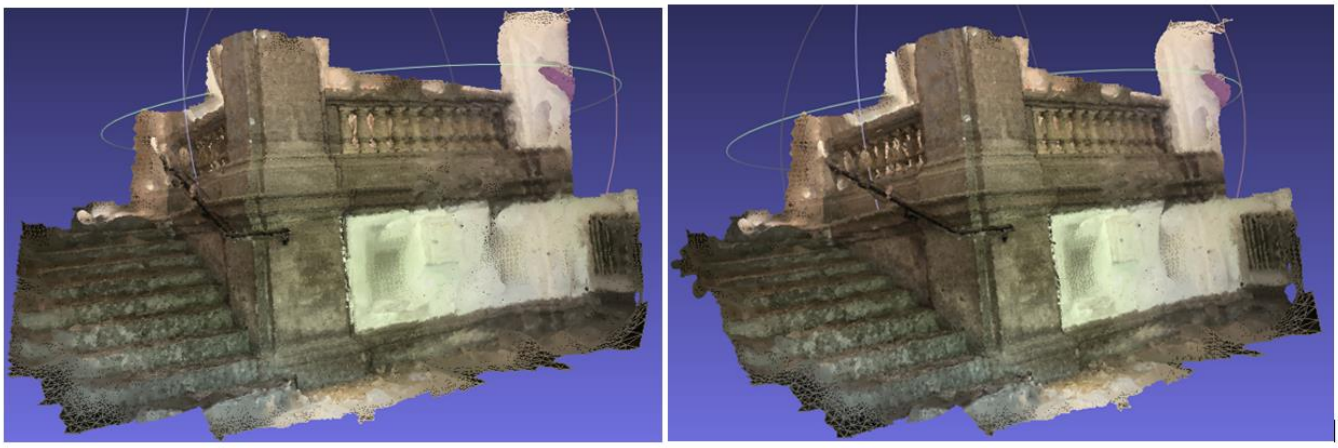

Figure 15: Object 2 in Meshlab