Summary for order 25_11_2020

Request

Create python scripts for metashape professional. In particular:

1. A script that counts the total number of markers detected in all of your photos
2. A script that applies different masks for different sets of photos.
3. A script that cuts point clouds with constant dimensions and automatically rotates them flat
4. A script that calculates the difference between two meshes

Resolution

The following points describe the process summary for the resolution of the presented problems.

1. Firstly the **marker_counter.py** script was created. This script worked but was slow. The buyer found a quicker way to count the total projection of markers in the chunk and so it was adopted in the **marker_detector.py**. This file has only been partially modified by the seller to be more compact and more efficient. It was implemented the possibility of importing automatically the GCP coordinate file but for unknown reason it wasn't working for the buyer so it has been commented out.

2. The mask.py script worked as intended after some adjustments.

3. The **region_size.py** script worked as intended after some adjustments. This scripts works only for properly georeferenced chunks so it cannot be applied everywhere (e.g the 1:8 dataset).

4. In the metashape version 1.5+ there is a function that calculates the DEM of a given mesh or dense cloud. This function then can be used to rapidly calculate the difference between two DEMS and so between two meshes. This process can be easly done without the need of writing a script.

The scripts can be found on the github page:

https://github.com/gbene/Python-orders

and downloaded with the link:

https://minhaskamal.github.io/DownGit/#/home?url=https://github.com/gbene/Python-orders/tree/main/25_11_2020

Final workflow

We can summarize the developed metashape workflow as follows:

1. Import images.
2. Apply masks (via **mask.py**).
3. Detect markers.
4. Check if in total there are 135 markers projections (via **marker_detector.py**).
5. Import GCP reference file.
6. Align photos (suggested medium quality).
7. Cut the point cloud (via **region_size.py**).
8. Build dense cloud (suggested with medium quality).
9. Build mesh (suggested with high quality).
10. Build DEM with mesh as source.
11. Calculate difference with DEM tools.

This workflow is resonably fast and has the possibility to batch the processes to further automate the workflow. The seller needs to verify that is valid by comparing the metashape pipeline with the matlab pipline previously used for the 1:8 dataset.

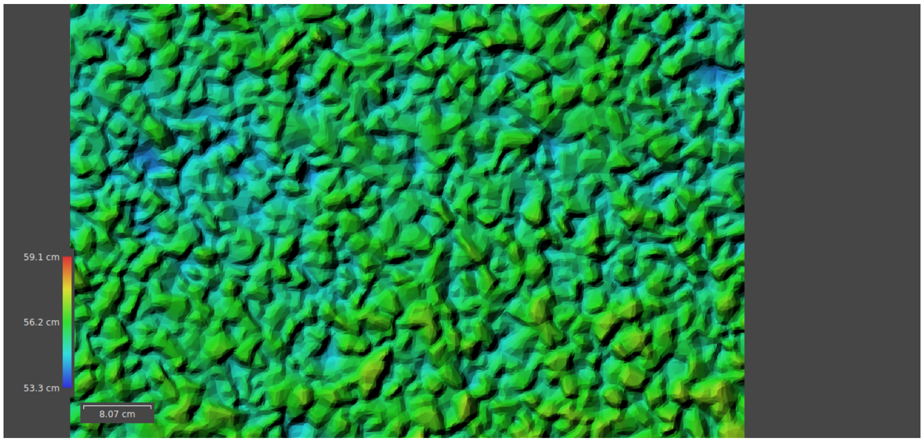

Figure 1: Example of DEM with a resolution of 0.5mm/px based on a mesh

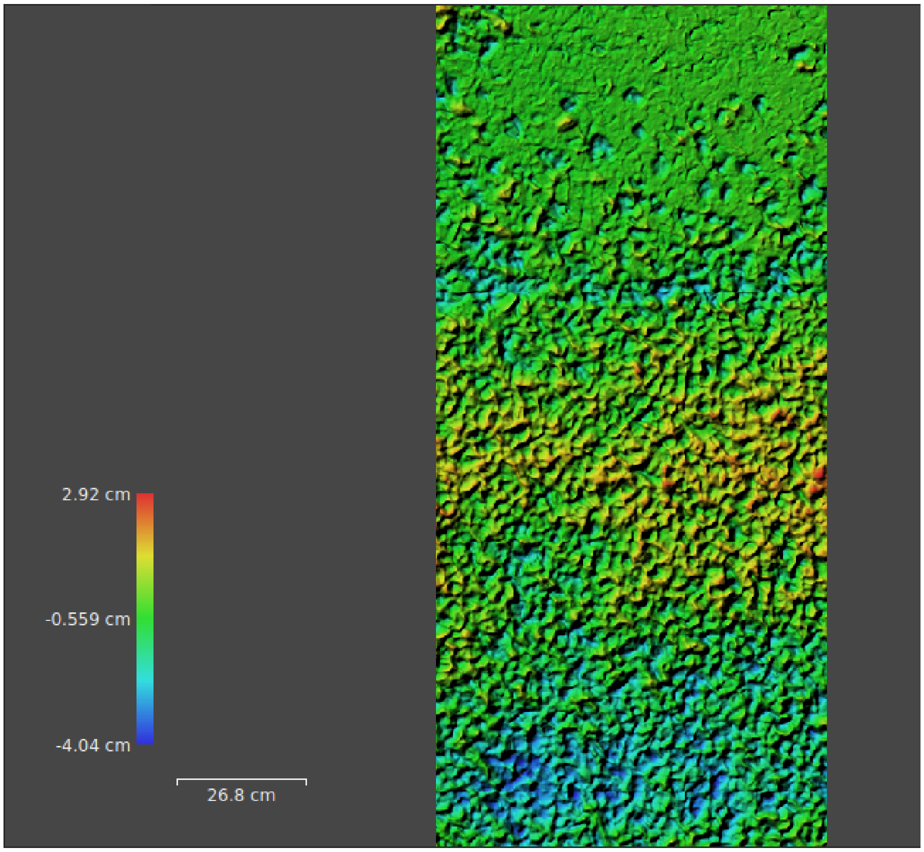

Figure 2: Example of DEM difference between the 2_1 and 2_10 dataset