BEAM Documentation

Contents

BEAM Installation Guide

Installing the BEAM tool simply requires users to run the install.bat file located in the BEAM project folder.

This script is a batch file that will install the dependencies for the BEAM tool automatically, creating an environment that can be used to run the tool. This environment contains the Python packages and dependencies needed to run the beam test program.

To use this installer, you need to have the following:

- A Windows operating system (64-bit)
- An NVIDIA graphics card with up-to-date drivers
- PowerShell installed and enabled (should be installed by default on Windows systems)
- An internet connection
- The default folder structure for the BEAM tool

The script will perform the following steps:

1. Checks if Mamba is already installed on your system. Mamba is a package manager for Python. If Mamba is found, it will skip to step 4.
2. Downloads the Mambaforge installer PowerShell. Mambaforge is a distribution of Miniforge that includes Mamba. The installer will be saved as a temporary file in the %TEMP% directory.
3. Runs the Mambaforge installer in silent mode with the following options:
    - InstallationType=JustMe: Install Mambaforge only for the current user
    - S: Suppress any user interface or prompts
    - D: Specify the installation directory as %USERPROFILE%\mambaforge
    - After the installation, the script will delete the temporary installer file and check if the installation directory exists. If not, it will exit with an error message.
4. Initializes Mamba by using the mamba init command. This will modify your PATH environment variable and activate Mamba by default in your shell sessions. If this fails, exit with an error message.
5. Creates the beam environment. This will install all the packages and dependencies listed in the file into a new environment named beam. If this fails, exit with an error message.
6. Exits with a success message.

To run this script, you can either double-click on it or open a command prompt in the directory and type its name. You may need to run it as an administrator if you encounter any permission issues.

The installation will take some time, as the dependencies are more than 3GB in size.

After running this script, the BEAM tool can be run by following the steps in the “Test Guide”. The start_interface_browser.bat file will activate the beam environment and launch the tool.

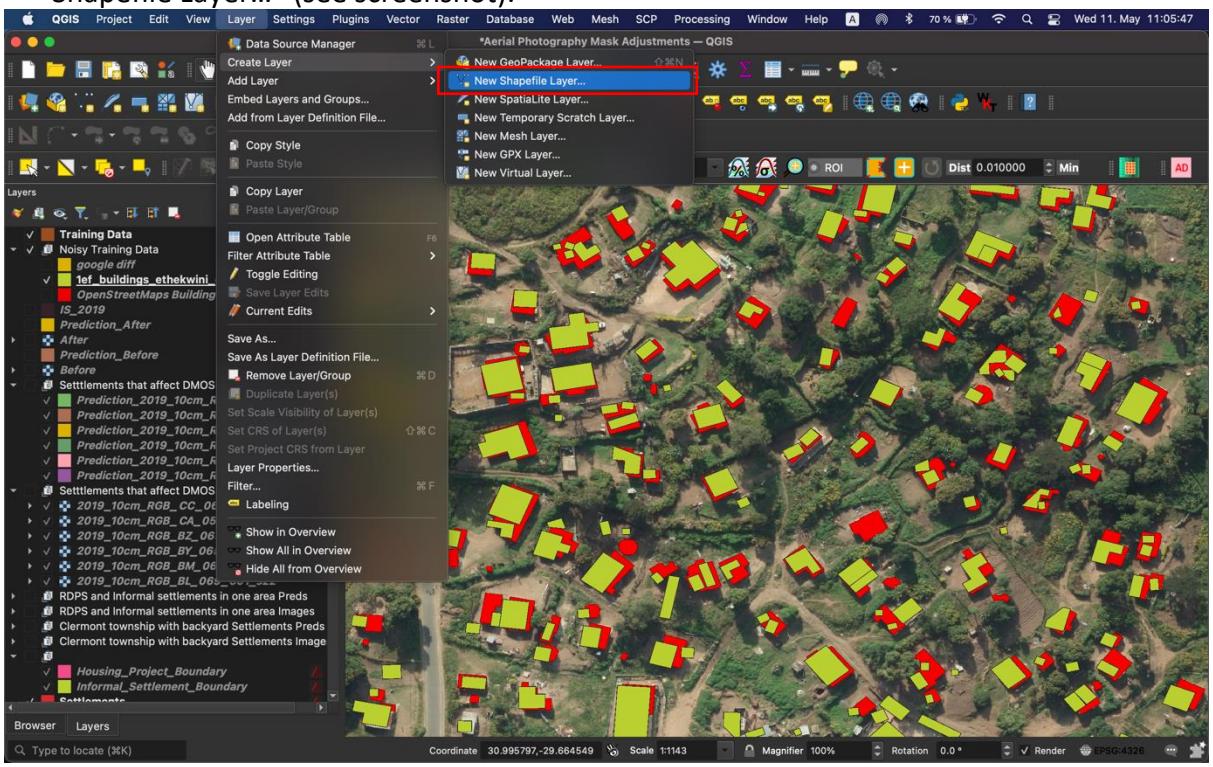
If firewall or network issues prevent downloading of the dependencies through the installer, the dependencies can be downloaded by referring to the version numbers in “environment.yml” (located in the “unitac-backend” directory).

Updates to the BEAM models can be downloaded by running the updater.bat file. This will check for changes to the source code in the UNITAC BEAM repository on Github (https://github.com/UNITAC-Hamburg/beam_update_2309) and download any updates.

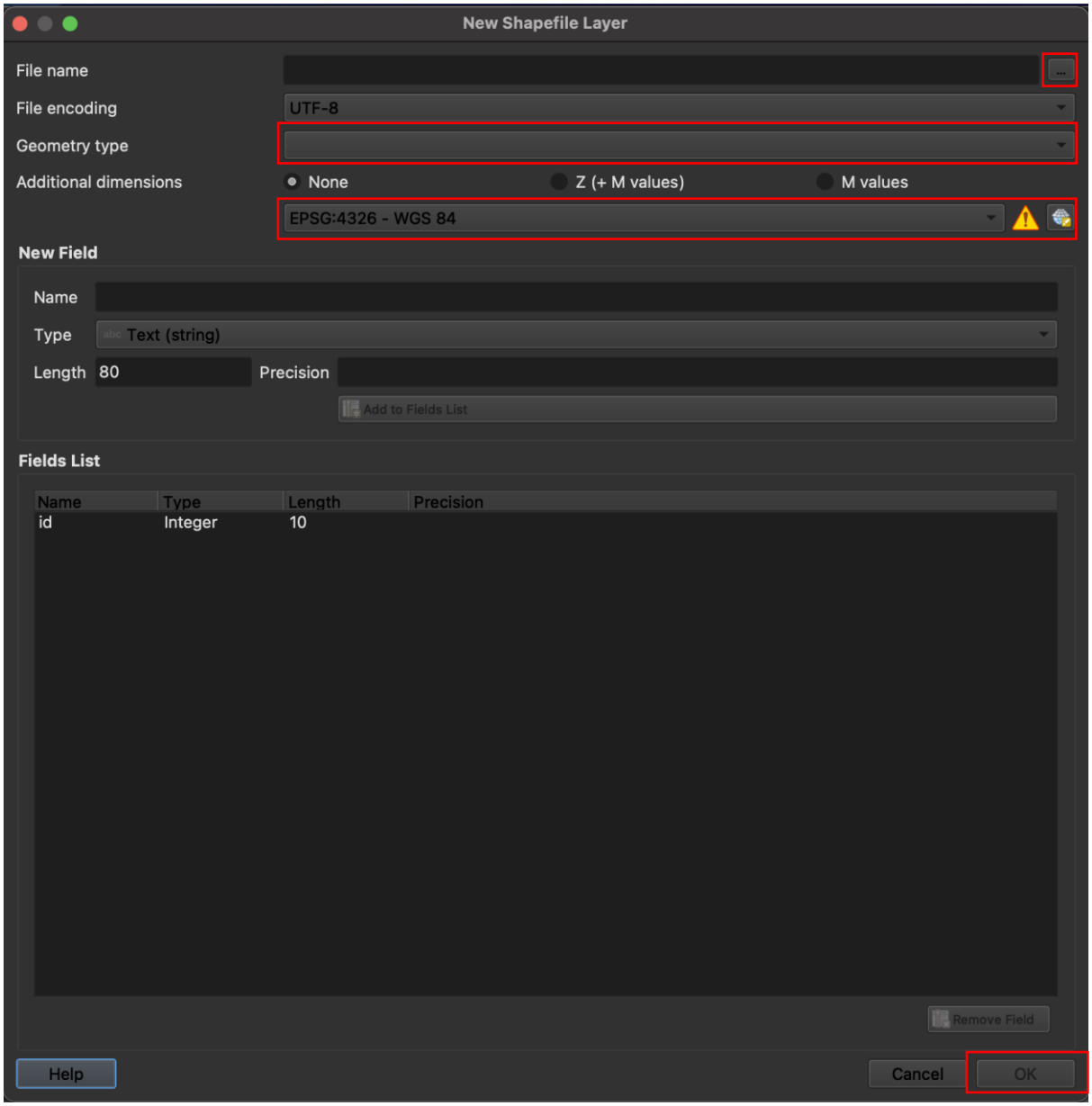
Label Creation

This section describes how to create new training data (labels) for the BEAM tool. The following section (labelling guidance) describes best practice and the guidelines to follow when doing so.

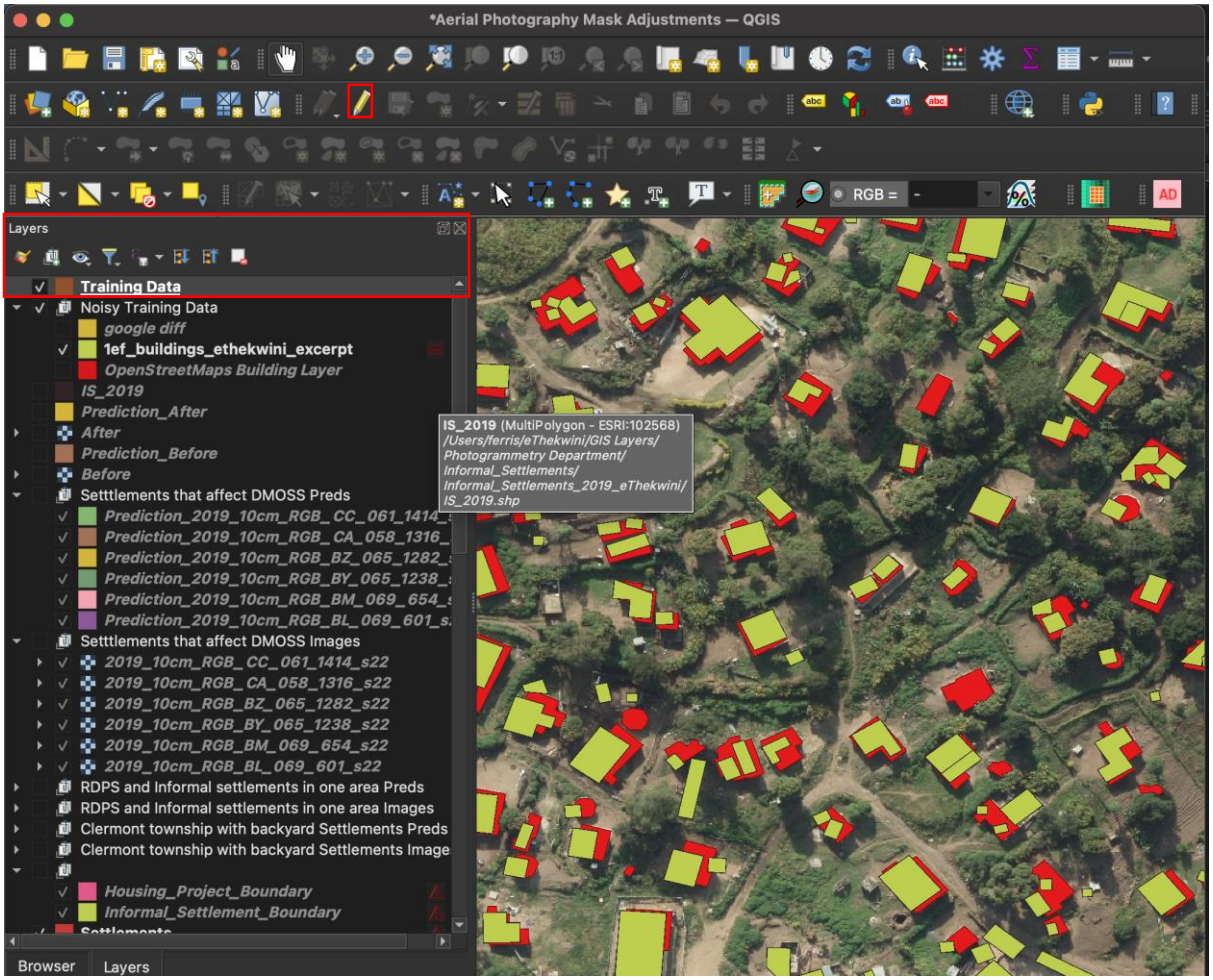
1. Open QGIS.
2. Load training images into the tool.
3. Select the “Layer” menu from the menu bar and go to “Create Layer” and then “New Shapefile Layer…” (see screenshot).



4. Now click on the three dots at the top-right of the window (see red box in screenshot below) and navigate to the folder where you would like to store the training data and give the file a name that you will later recognise. Click on the “Save” button.
   Now choose “Polygon” in the “Geometry type” dropdown menu and the coordinate reference system (CRS) that your aerial photography is in from the “Additional dimensions” dropdown menu. Finally, confirm your choices by clicking on the “OK” button at the bottom right.

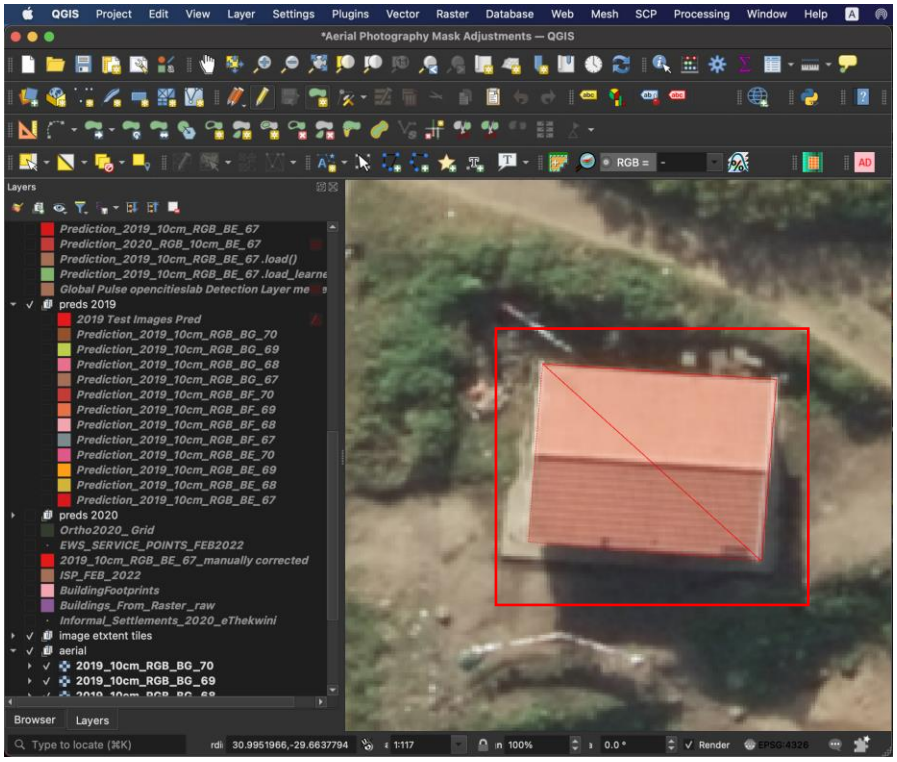


5. Now you will see a new layer with the name you chose in the “Layers” menu on the left hand side of QGIS (in this example “Training Data”. Click on the layer and click on the “Toggle Editing” button in the menu above (marked in red in the screenshot below). Now you should see more buttons from the QGIS menu being useable.

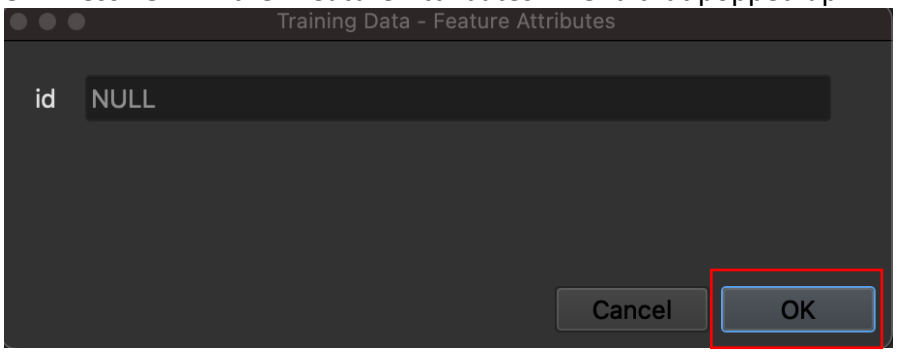


6. Click on the “Add Polygon Feature” button marked in red below to be able to start marking building pixels. Now click on the corner of a building you would like to mark and keep clicking into the other corners until the entire building is marked in red

7. Now click on the corner of a building you would like to mark and keep clicking into the other corners until the entire building is marked in red. To confirm your selection, right-click once.



8. Press “OK” in the “Feature Attributes” menu that popped up.



9. Continue marking all buildings in the images you would like to use to create training data (for eThekwini, one image of 10,000 x 10,000 pixels corresponding to one square kilometre on the ground was used).
10. Once you are done, click on the “Save Layer Edits” button in the menu (marked in red below).

11. You can now close QGIS and open the Python notebook used for tiling images and labels.

12. You will have to make changes under “3. Set Paths for Images and Masks”, where you need to adapt the directory for the “path”, “images_list”, and “shp_path” variables to match where you stored these on your computer. Lastly, you will also have to adapt the third parameter of the “save_masks” function under “Create Masks Corresponding to Selected Images” to match the file location on your computer.

13. You can now execute the script. The small image and label tiles will be saved in your chosen output folder and you can now use them to train the deep learning models.

Labelling Guidance

This section should be used in conjunction with the BEAM label creation guidelines (above). This is a supplementary section that serves to highlight best practice for label creation, and aims to provide a consistent set of rules to follow when creating training data for the BEAM model.

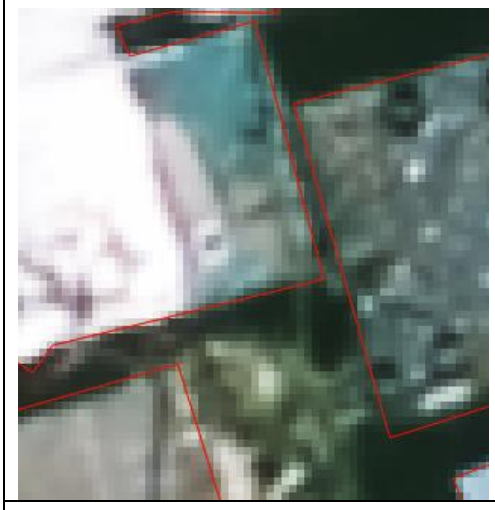
Overview

Generating accurate labels is important for the quality of the BEAM model and its performance. The aim when generating labels is to capture the entirety of each structure's rooftop using as few vertices as possible. The key point is to cover the entirety of the structure within one or two pixels of accuracy. In some instances, this is straightforward, and the most important consideration is simply tracing the rooftop of each structure accurately (to within one or two pixels of its edges):

	
Correct	Incorrect

However, in other cases, particularly in complex scenes or areas with high levels of informality, it is not always clear where one building starts and another one stops, or what the true shape of a building is if it is covered by overhanging trees or shadows. In these situations, it is essential that:

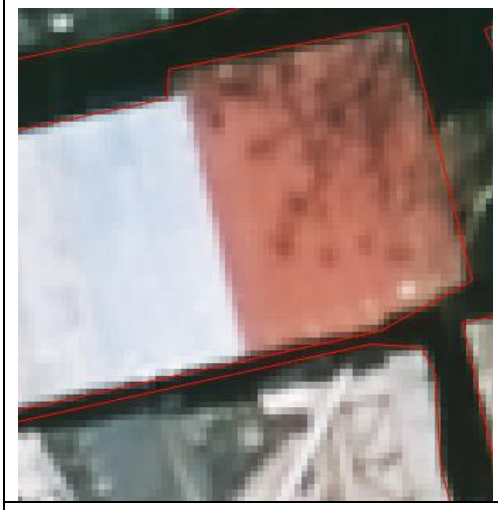
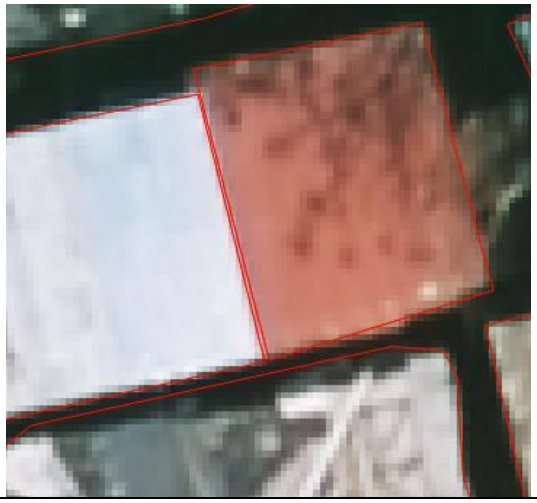
- Polygons for buildings that are close together do not overlap

	
Correct	Incorrect

- Building shapes are not compromised by the presence of trees or shadows (this can mean inferring the shape of a building)

	
Correct	Incorrect

- Every effort is taken to ensure that individual structures are correctly discerned. This can mean cross-referencing different sources of imagery or closely inspecting structures to discern where gaps or shadows are visible. Particularly in informal settings, this can be confusing as large structures are often built from several different types of roofing material. In this situation, it is important that the structure is captured as a single polygon.

	
Correct	Incorrect

Finally, labels should accurately capture building shapes without being overly prescriptive about jutting facades and other irregularities that will complicate polygons and increase the number of vertices used:

	
Correct	Incorrect

In general, the following guidance applies:

- Avoid modifying the labels to accommodate shadows, trees, or other image-dependent features.
- Label only the visible rooftops of buildings in off-nadir imagery, and avoid capturing building facades, windows, and so on
- Use additional image sources for added context and detail, but avoid using outdated or inaccurate sources to infer anything that is not visible on the image itself

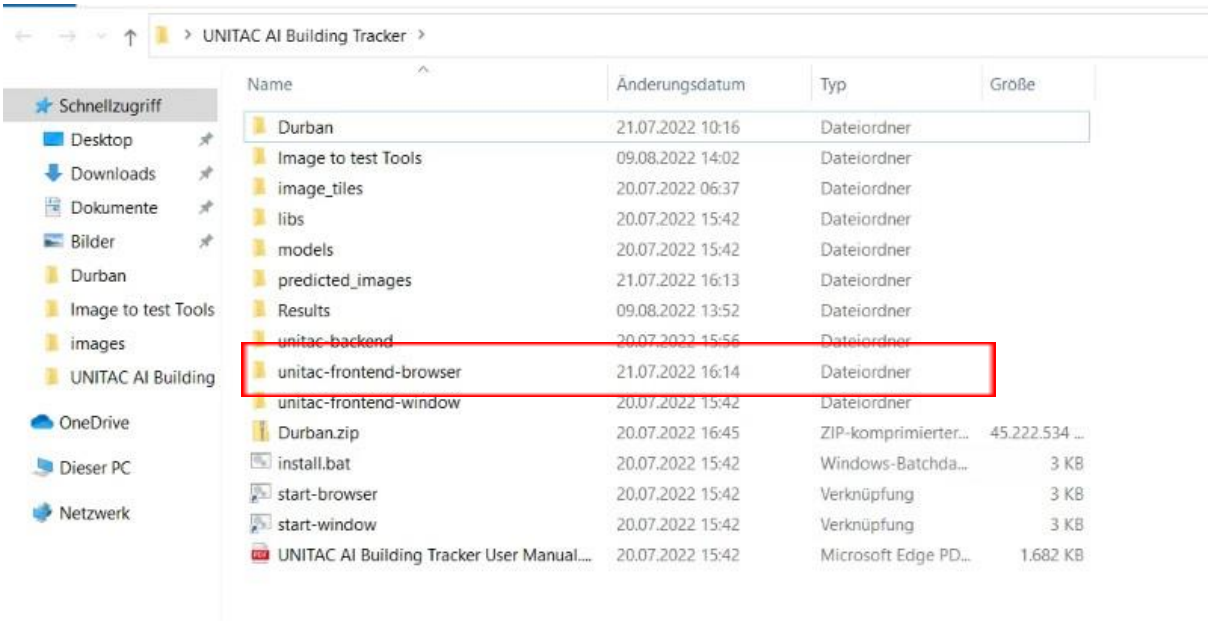
Test Guide

How to run the tool locally.

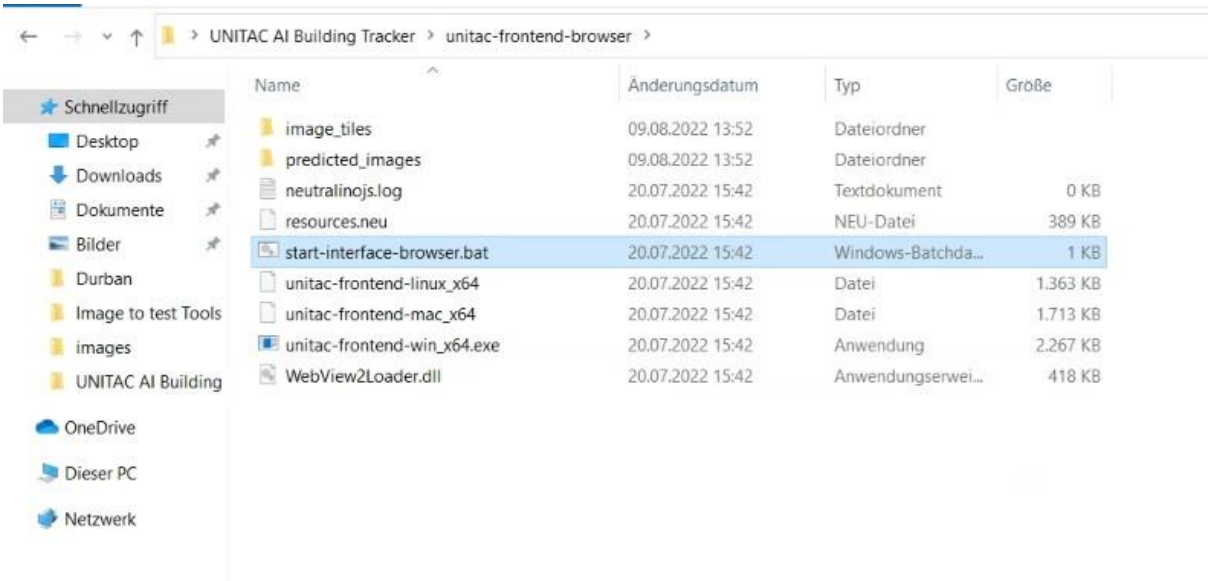
Steps:

1. Open the folder where you have saved the tool and its subfolders.

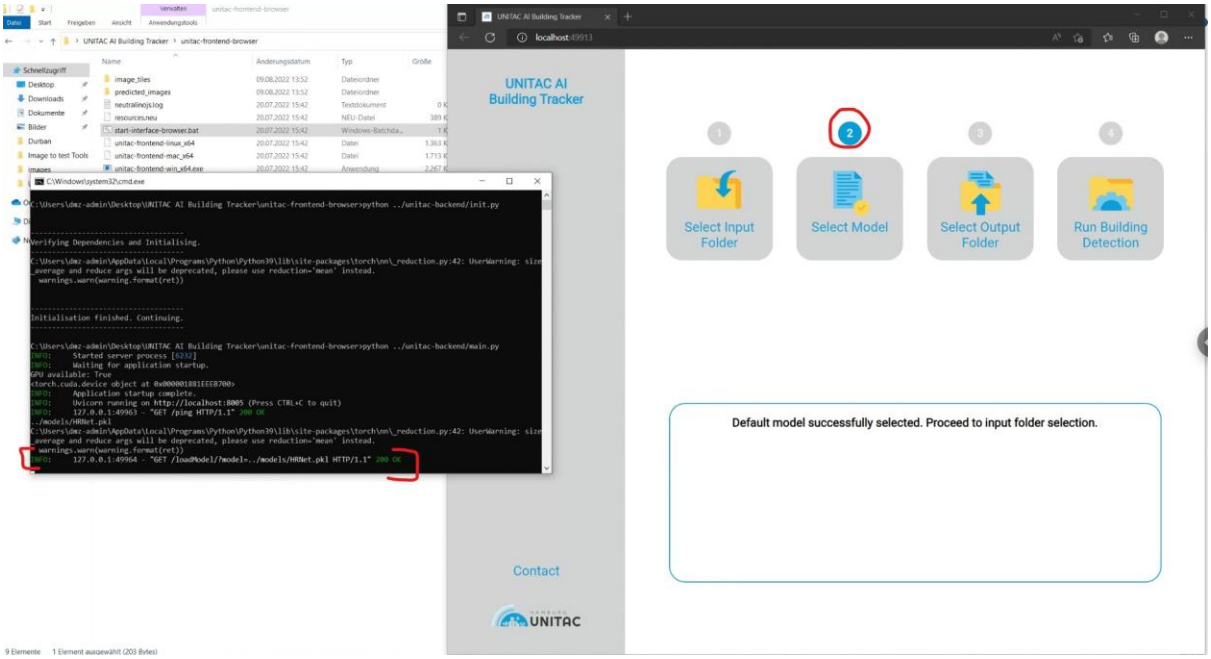
2. Open the **unitac-frontend-browser** folder



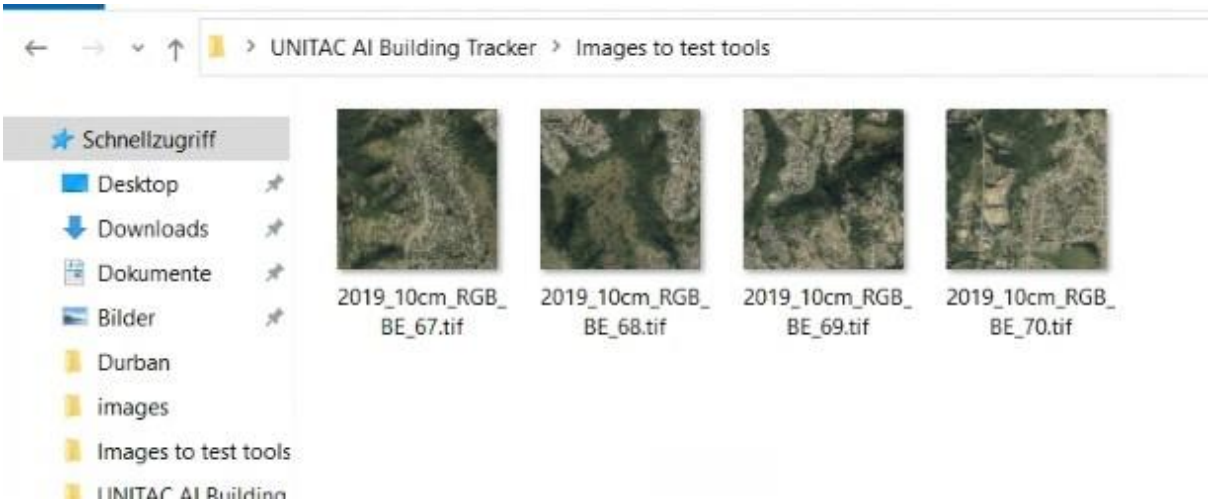
3. Double click on **start-interface-browser.bat** file.



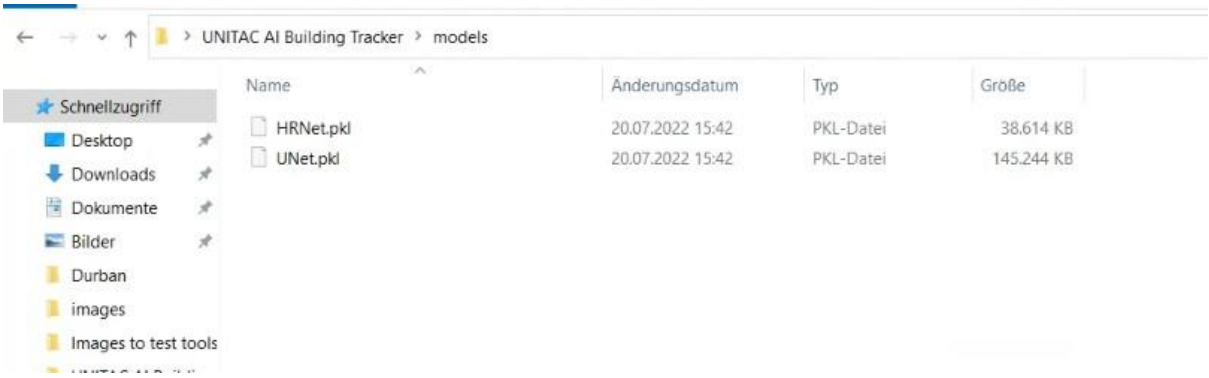
4. Two windows should open. The first one is the **UNITAC AI Building Tracker** (Interface) and the second one is the **command console** (Black window). Wait some seconds until the console finishes the initiation and runs some lines that end up with **200 OK in green.** Once that is finished, you will see **above of the button "Select Model"**, the number two is going to change to **blue**. This means that the pre-selected model is loaded.



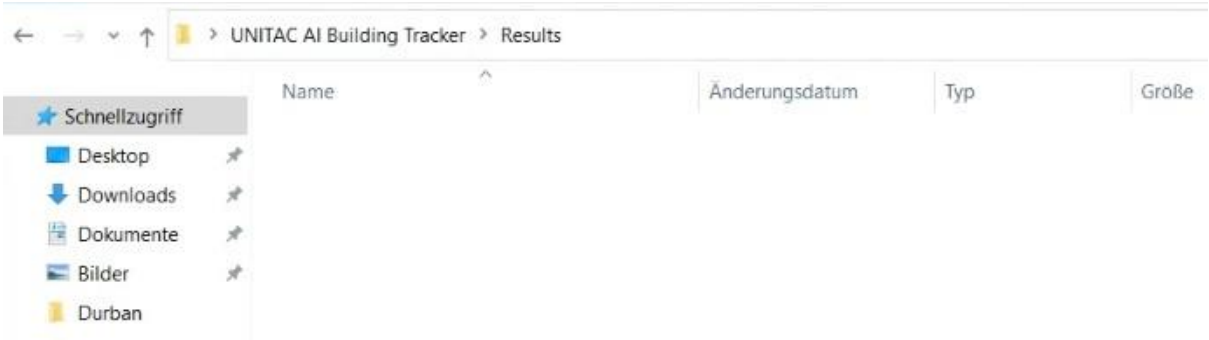
5. Now we have to select the **Input Folder** with the *images* where the model will detect the structures. Click on the **Select Input Button** and choose the folder where the images you would like to run are saved**.** Once you select the folder, the number above the button will turn blue.



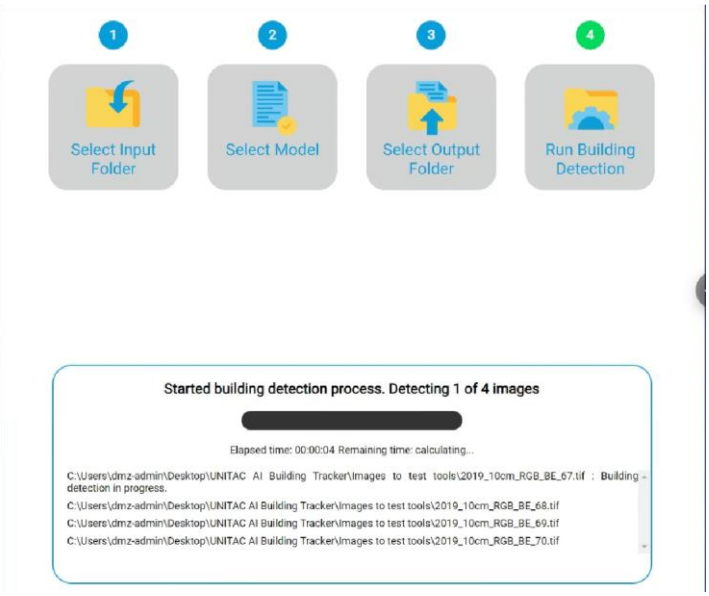
6. For the demo we are not going to change the model but if you want to you can press on the button where it says **Select Model** and look for other models on the **models** folder to select the one you like.



7. Now we have to select the **Output Folder**. Here the model will create a shapefile (.shp) with the outlines of the building structures. For this we are going to select the folder **Results.**

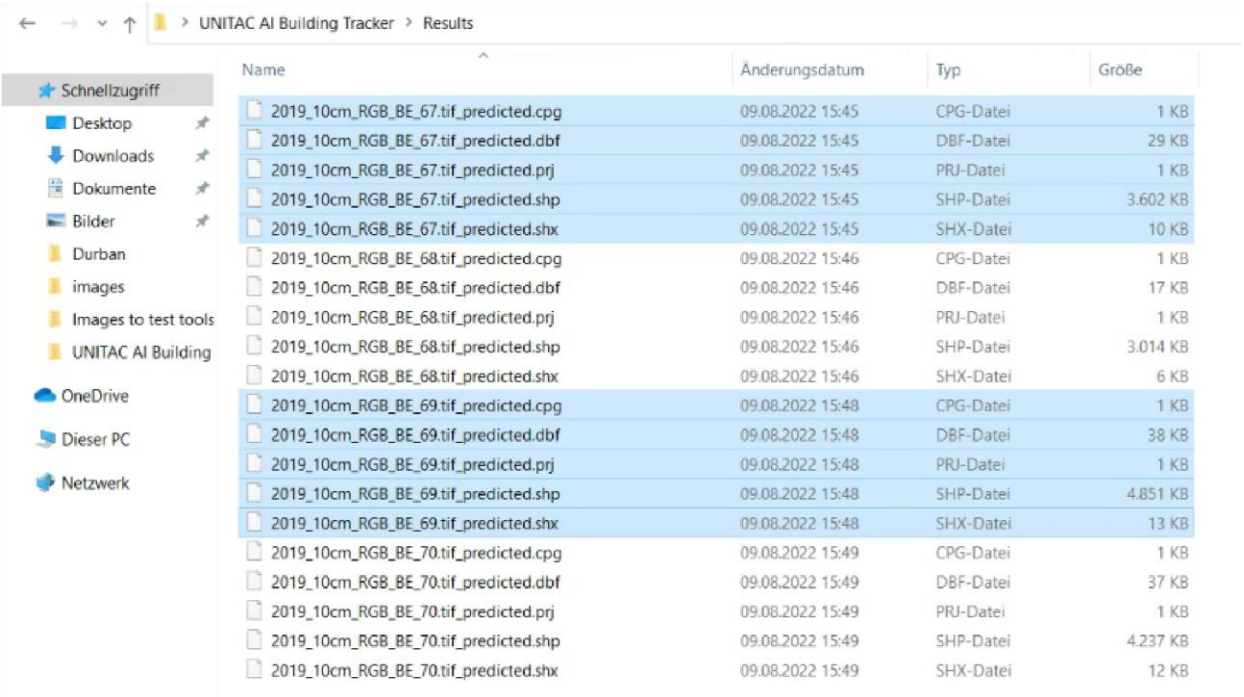


8. **Run Building Detection button!** Now the tool will start to analyse the images. You are going to see that the button **turns green** and we can see on the messages below that says **building detection in progress** on the first image.

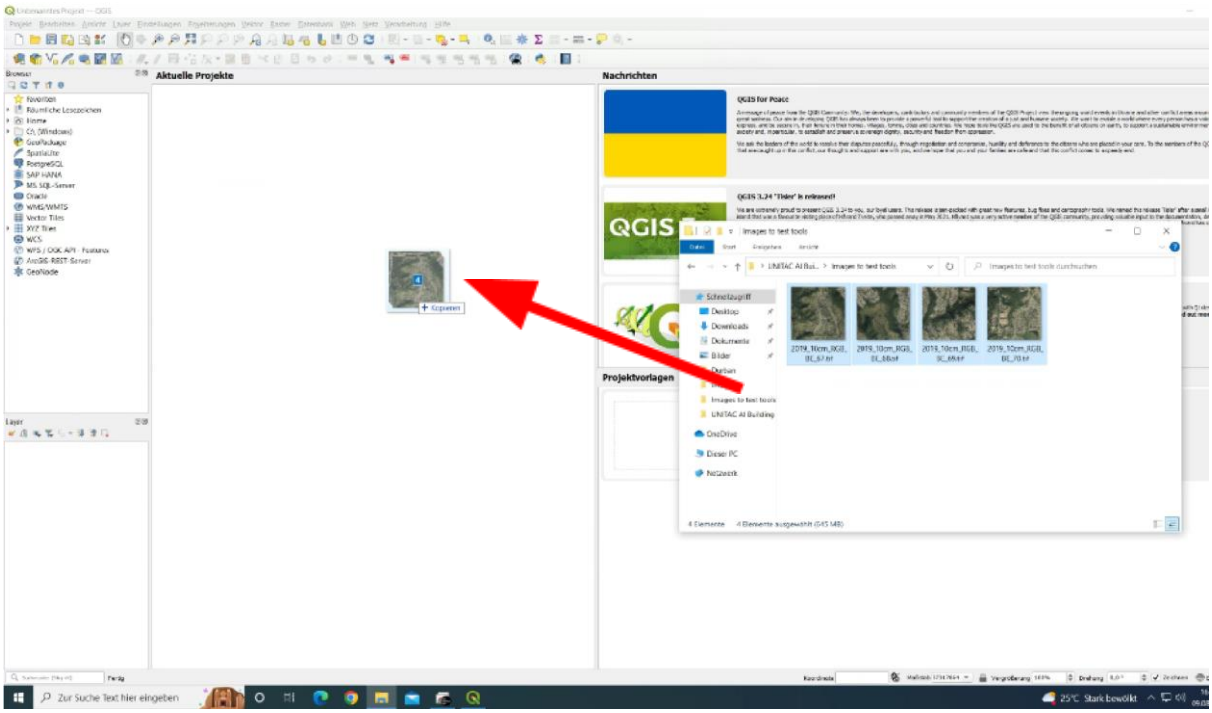


9. It should take around 1-3 minutes per image depending on your computer. It will go one by one till it finishes everything. It will say **Building detection completed. Shapefiles were stored in the output folder and can be validated now** when it done.

10. We can close the Building Tracker and now we have to go to our **Results** folder to see the **shapefiles**. **Each shapefile has 5 files**.

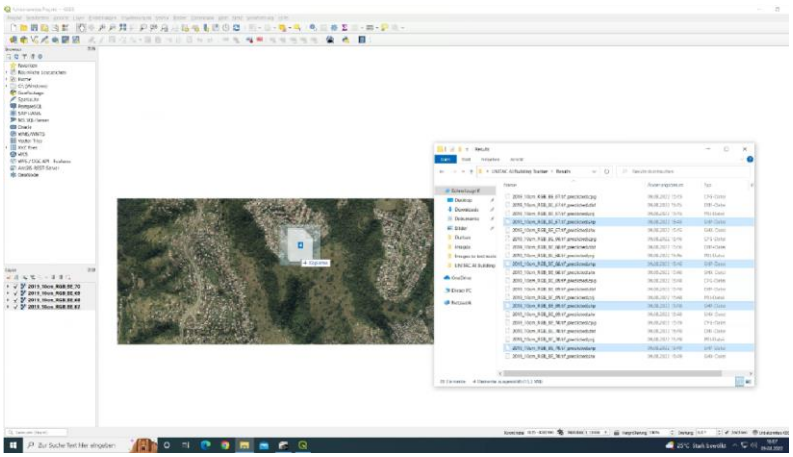


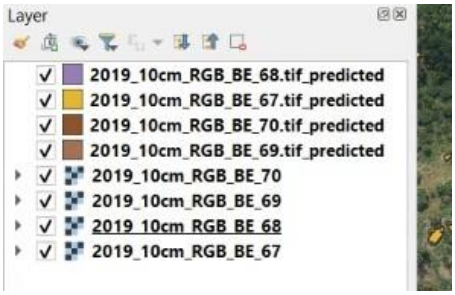
11. In order to be able to **visualise the images and the shapefiles** we are going to open **QGIS.** You can look it up by looking for QGIS on the windows start panel. We are going to drop the images that are on the **Images to test tools** folder into QGIS.



12. You are now going to see the 4 images loaded on the **Layers panel.**

13. Now we are going to go to our **Results** folder and select the four files that end with **.shp** and drop them on top of the images. Make sure that the predicted Layers (Shapefiles) **are placed above the images** in order to see it.





14. Now you can see the model predictions.

15. If you want to **change the colour or the format** of the shapefiles you can do it by right clicking on top of the layer and selecting **Setting** and selecting the label **Symbology**. Here you have some predetermined formats. Choose one and press **OK.**

16. Now you should be able to see it better. [Blue outline]