

Blender addons

ESRI Shapefile import/export and georeferenced raster import

This blender addon is a collection of 4 tools:

- [ESRI Shapefile importer](#)

- Import point, pointZ, polyline, polylineZ, polygon, polygonZ feature class. Actually cannot import multipoint, multipointZ, pointM, polylineM, polygonM and multipatch.

- Polygons are created with the new blender bmesh module that allows Ngons.

- Can handle multipart geometry but actually if the part defines a polygon hole, a face will still be created.

- Can use data contain in dbf file to define Z elevation values. You need to know & manually enter the name of the field that contains the values (don't worry about case sensitive).

- For polygon features only, it can uses data contain in dbf file to define Z extrusion values. You need to know & manually enter the name of the field that contains the values (don't worry about case sensitive). Note that the extrusion can be very slow with lot of features but you can see progress output in Blender console.

- Can automatically increase the grid size and viewport clipping distance if necessary (think to center 3dview with shift+C after the import).

- Georeferencing management: because Blender (and most of 3d software) cannot strongly handle objects that are far away scene origin, and because coordinates values are limited to 7 significant figures, it's necessary to create the mesh near to the scene origin. For avoid georeferencing lost, all objects created with the script contains 2 custom properties which represent the shift values operate in X and Y axis. When you try to import a shapefile, if the scene contains one or more object with delta X/delta Y custom properties, you have the possibility to consider one of them to adjust the position of the new imported object.

Of course if you move the object or edit the mesh, delta X and Y properties will have no meaning. Optimal workflow requires you import all your useful data with correct shift adjustment in a first step and then you can break georeferencing by moving/editing the mesh or object.

Troubles:

This script depends on [pyshp](#) library that is in beta version, so sometimes it cannot process the shapefile. Typically, if you got "Unable to read shapefile", "Unable to extract geometry" or "Unable to read DBF table" error, these are pyShp issues. In this case:

1. Try to check if there is any update of pyshp lib (<http://code.google.com/p/pyshp/downloads/list>). Note that the distributed version with this script is v1.1.4 instead of v1.1.6 actually available because I got fewer troubles with this one.

2. If there is no update available or if update doesn't correct the problem, try to open and re-export the shapefile with any GIS software. You can also try to clean/repair geometry, delete unnecessary fields...

For polygons import, if you see faces which seem to be strangely filled try to remove duplicate vertex on the mesh (modify tolerance distance if necessary)

- ESRI Shapefile exporter

- Can export a mesh to pointZ, polylineZ or polygonZ shapefile. If the mesh has georef data then this will be considered.

Note that Blender cannot handle attribute data include in dbase file linked to the shapefile. So if you want import a shapefile for edit it into Blender and then re-export it, you will, of course, lose attribute data.

- Georeferenced raster importer

- Import common image format associated with a [world file](#) that describes the location, scale and rotation of the raster in a geographic coordinate system. The world file extension can be *.wld or must be respect one of these conventions: tif --> tfw or tfwx or tifw, jpg --> jgw or jgwx or jpgw, bmp --> bpw or bpwx or bmpw, png --> pgw or pgwx or pngw ...

- 4 import modes: on a plane mesh, as background image for orthographic view, as UV texture mapping on a georef mesh or as DEM for warp a mesh with the displace modifier.

- If the raster world file has rotation parameter you can still correctly import it on a plane or on a mesh but not as background.

- If you import on a plane, the new mesh will have the 2 custom properties which represent the shift values operate in X and Y axis, and it can be used as reference for another shapefile or georef raster import.

-For DEM import you can use GDAL tools (if installed) to process large file and any GIS raster data format.

Note: if the imported image appeared fully black, try to config. the light according to the scene (for example try the sun light type)

- Georef properties manager

For manage the delta X and delta Y customs properties, 2 operators are available in object properties panel:

- *Update georef delta* operator can be used to copy delta location properties from another georef object to an active georef object and to update the mesh position.

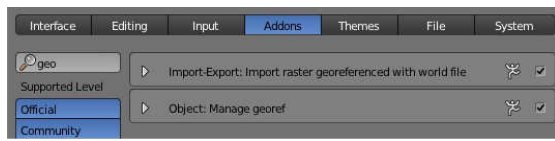
- *Create georef delta* operator can be used to calculate delta location properties for an active ungeoref object by reference to another georef object.

Install

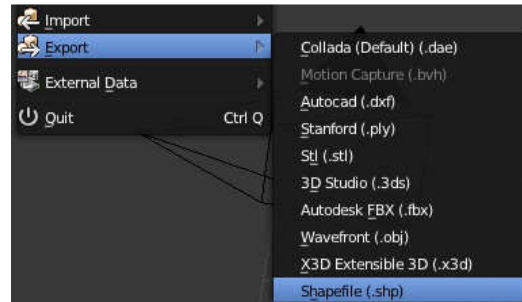
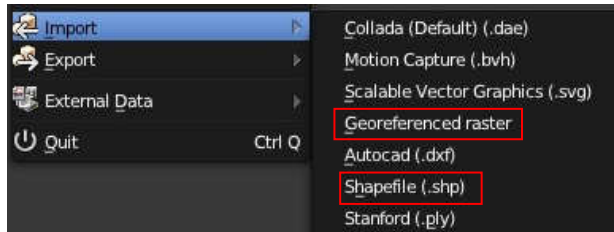
Just copy paste the files bellow to *C:\Program Files\Blender Foundation\Blender\2.65\scripts\addons*.



Don't forget to unble the 3 addons in Blender user's preferences

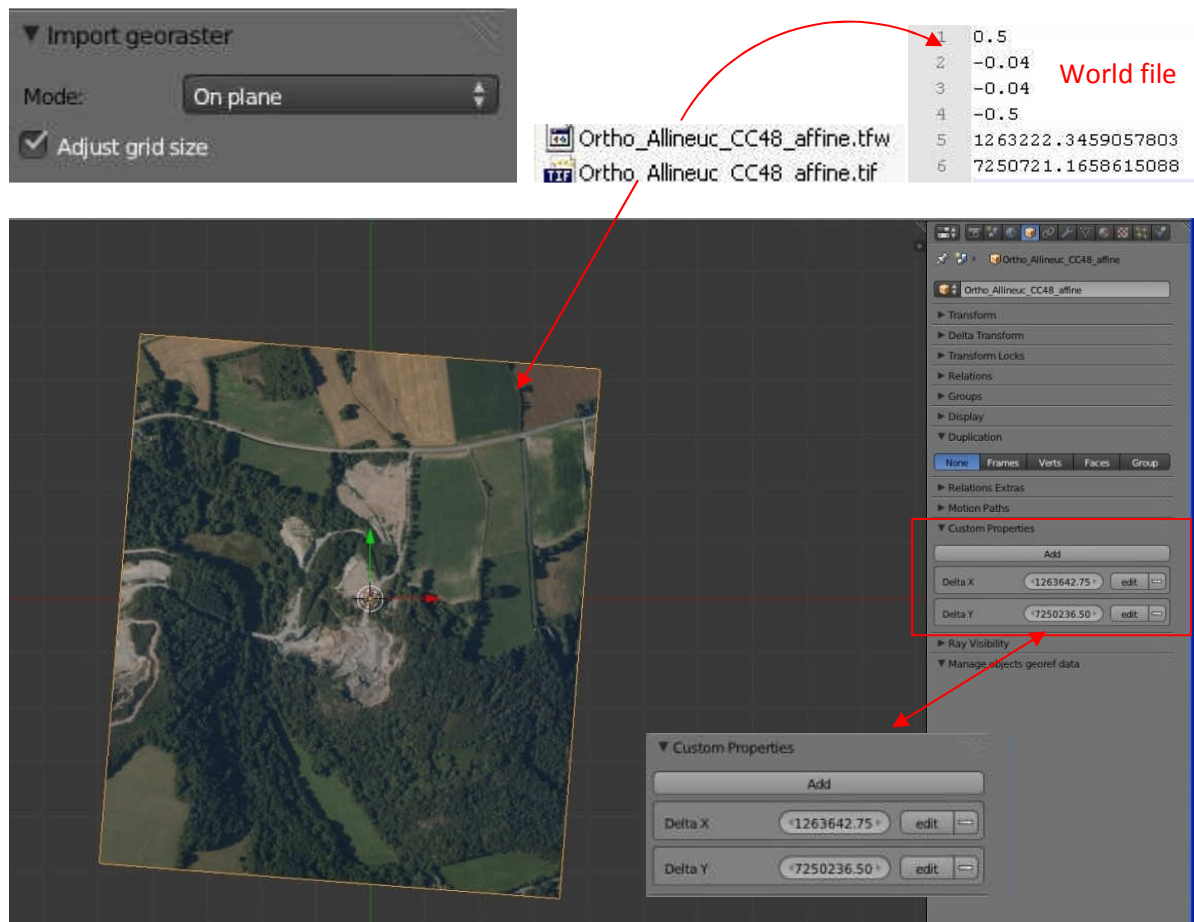


Usage



Workflow example

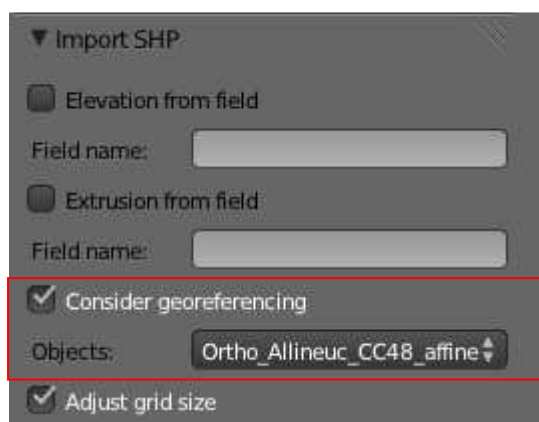
First, we can import a georef raster on a plane:



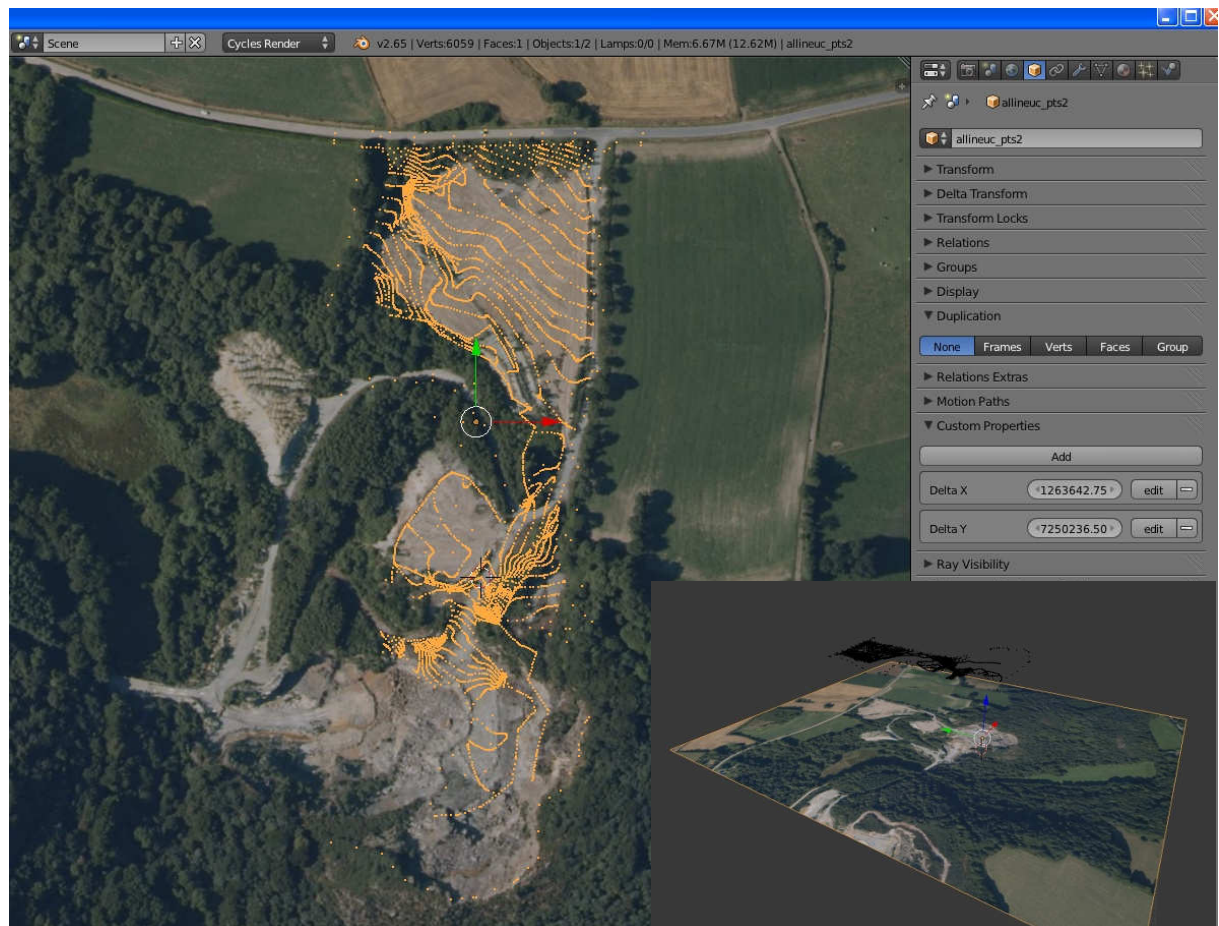
As we can see, the script takes into account the world file rotation parameters if necessary.

In the object properties panel there are 2 custom properties which are the XY deltas used to shift the mesh.

Now we can import some shapefile:

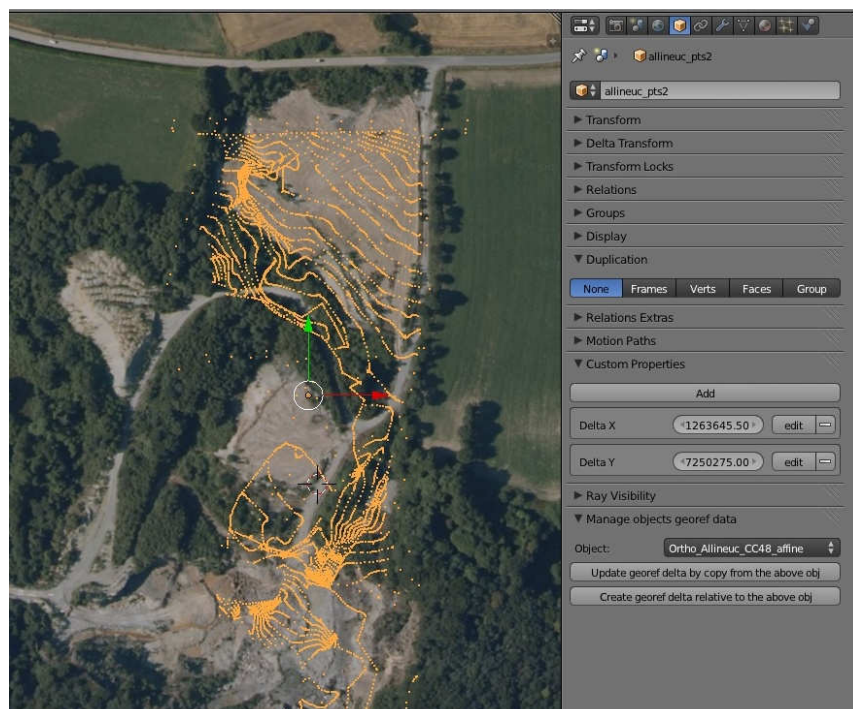


The script detects that there are some georef objects (objects with custom properties), and the user can use these objects as reference to his next import, so the data will be correctly shifted.

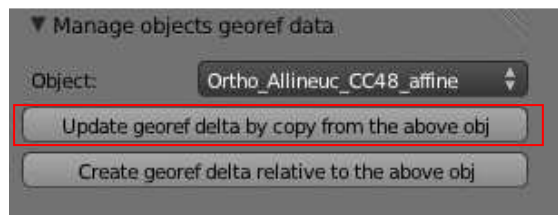


The superposition will be correct only if the geographic projection system is the same. Note that one Blender unit = one map coordinates system unit. So, units depend on your data.

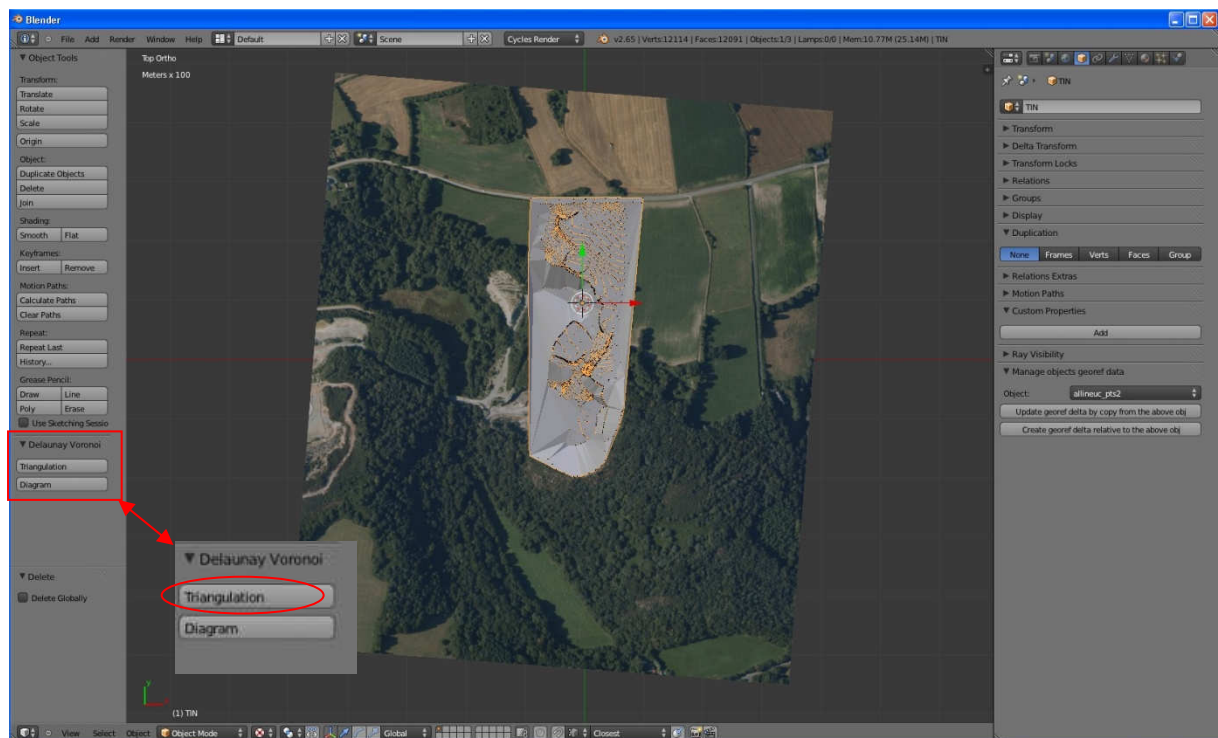
If we undo and redo the import without consider georef data, the object position will not be correct:



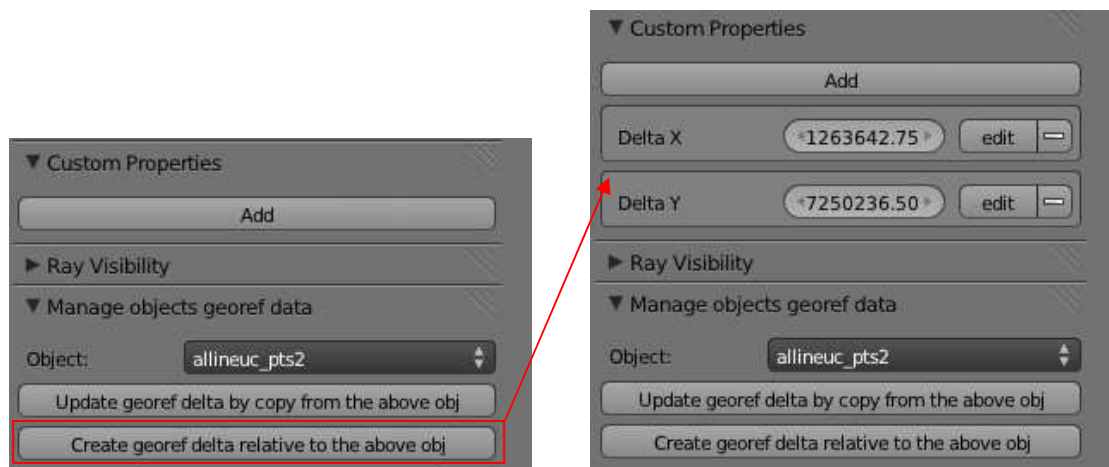
But in object panel there are 2 operators to manage georef deltas. With the first we can easily readjust the mesh to the raster.



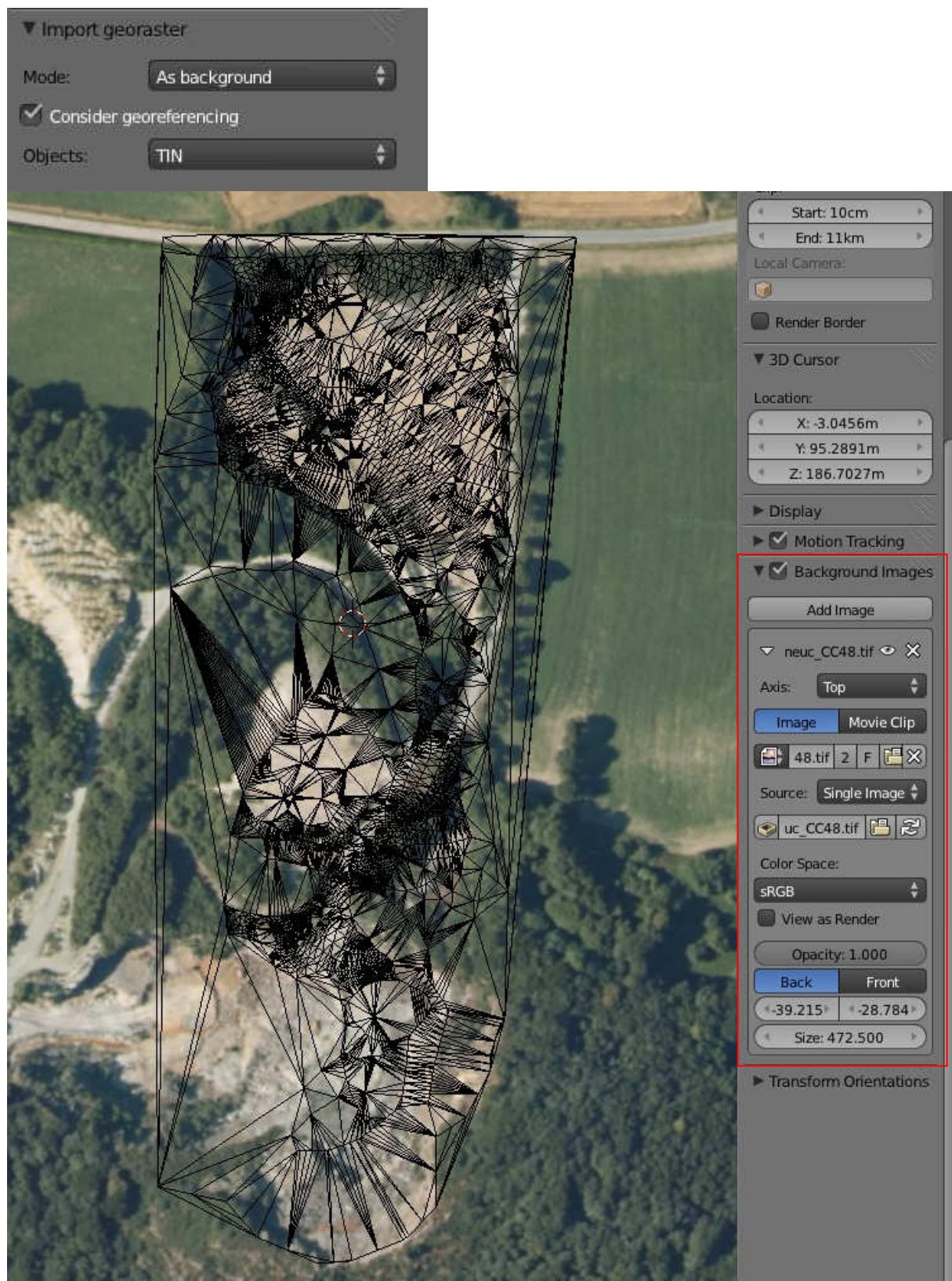
Now with my [Delaunay triangulation script](#) I can triangulate this point cloud to get a terrain mesh:



This new mesh hasn't georef data but we can create them:

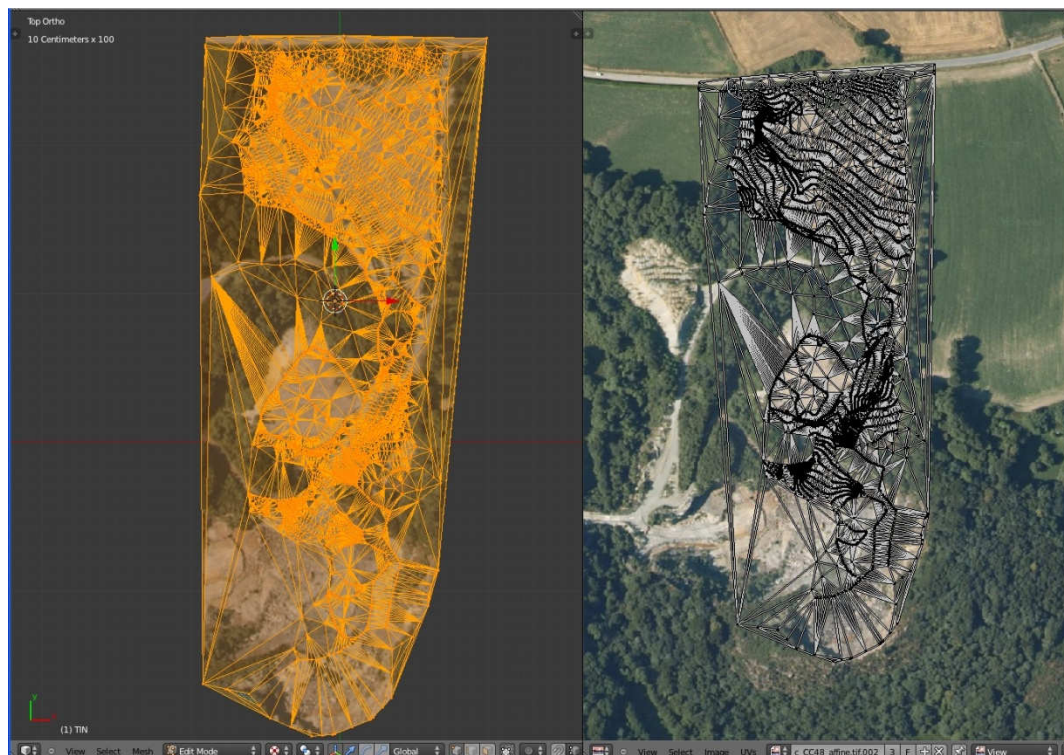
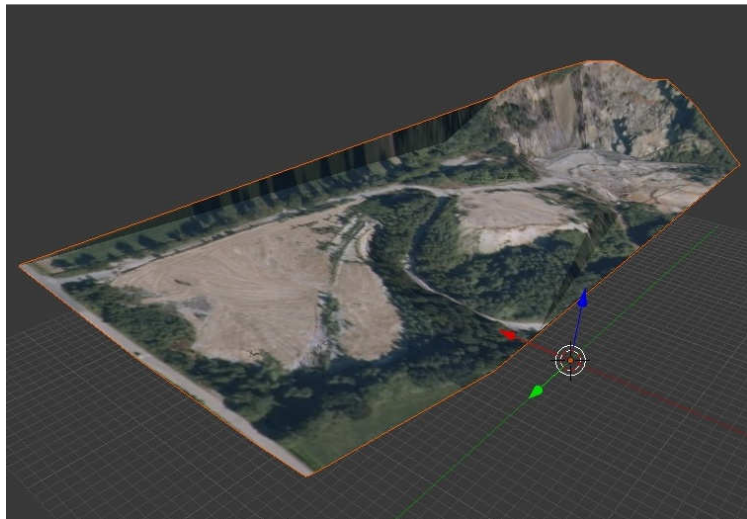


To view the tessellation we can delete the point cloud and the raster, toggle wireframe shader and re-import the raster as a background image:



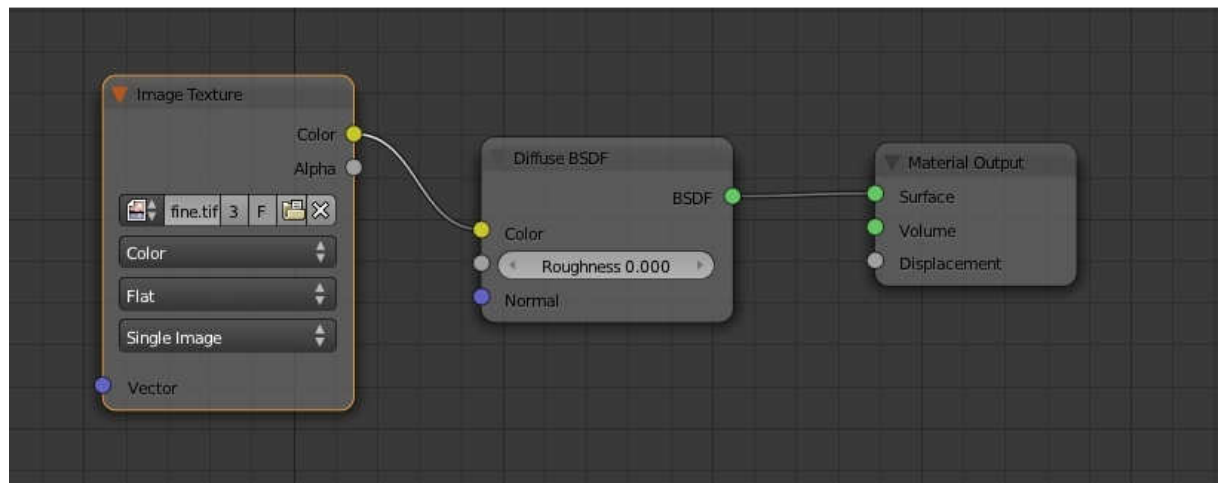
Note that you cannot import as background a raster that needs a rotation, and you cannot use a background raster as georef data reference for another import.

Finally, another possibility is to automatically UV map the raster on the TIN:

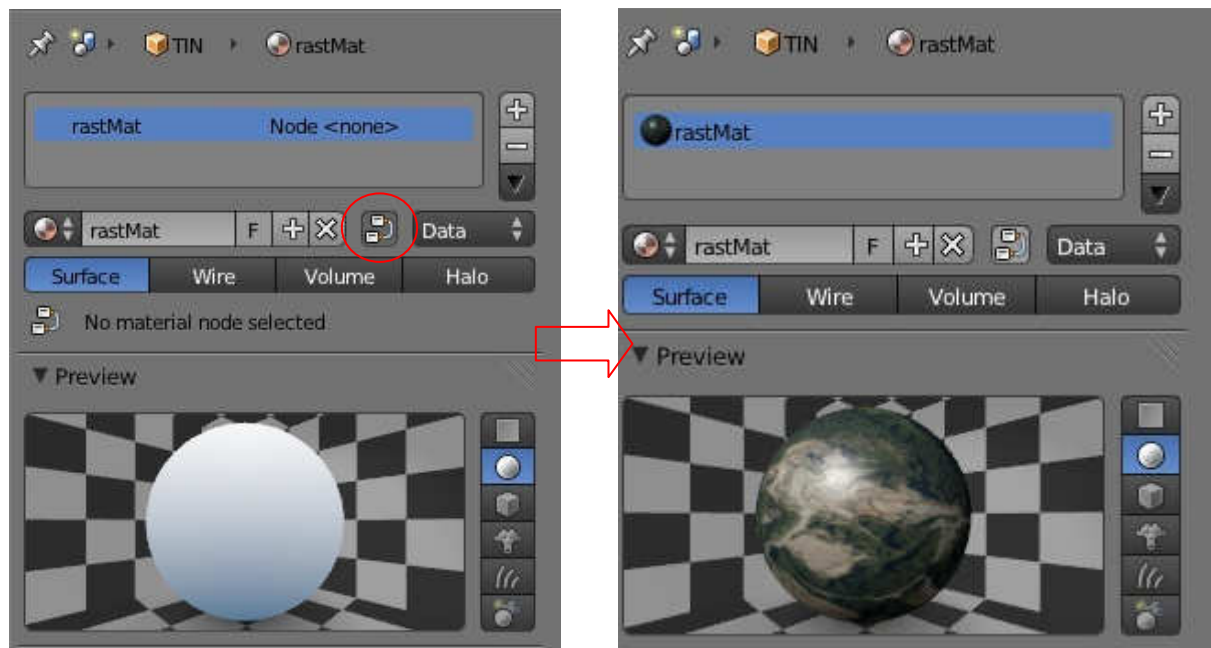


Here the UV Mapping

If you want to render the raster texture, note that the script config Cycles shader nodes as follows:



It also configures material for Blender internal renderer. In this case you need to disable the shader node if necessary:



Also, because the scene can be very large, don't forget to configure camera clipping distance according to the scene. If you still see black faces error after setting *clip end distance* try to set the *clip start distance* closer to the scene, it will help Blender to improve vertex position according to the Z depth of the camera.

Using DEM GRID

If you want use DEM GRID data source to warp/drape an imported raster, there are 2 strategies:

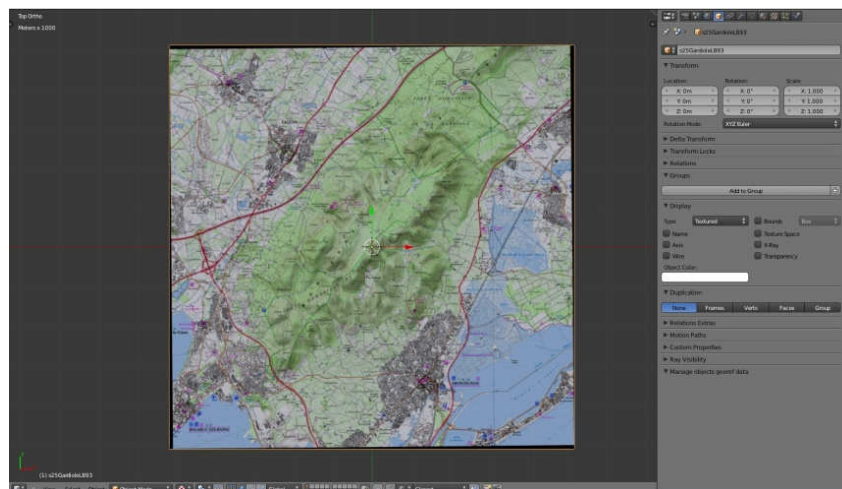
1. With any GIS software convert your DEM into point features shapefile that you can import and triangulate in Blender.
2. Use the displace modifier with a DEM raster. In this case, the script can help you to config the displacer.

Usually, a DEM raster can be coded into different bit depth depending on the data accuracy:

Bit depth	Range of values that each pixel can contain
1 bit	0 to 1
2 bit	0 to 4
4 bit	0 to 16
Unsigned 8 bit	0 to 255
Signed 8 bit	-128 to 127
Unsigned 16 bit	0 to 65535
Signed 16 bit	-32768 to 32767
Unsigned 32 bit	0 to 4294967295
Signed 32 bit	-2147483648 to 2147483647
Floating-point 32 bit	-3.402823466e+38 to 3.402823466e+38

For the moment, this script can only process 8 or 16 bit raster because it seems the displacer cannot correctly deal with 32 bits integer or floating DEM.

For use a DEM, fire on Blender and import a map raster on a plane:



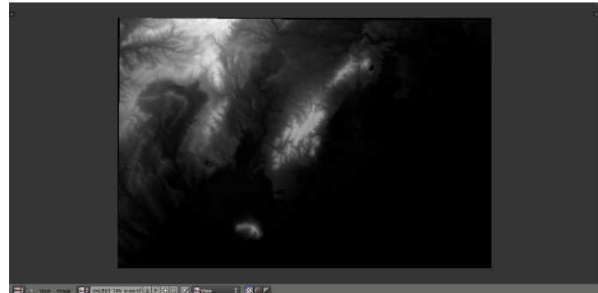
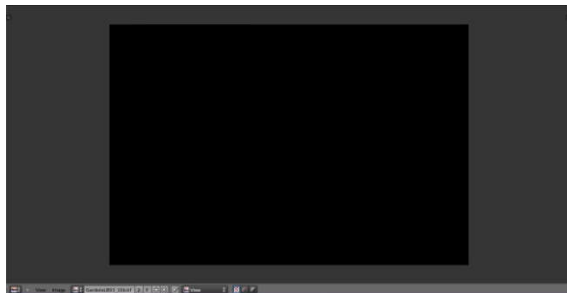
Then import the height map with the *As DEM* option:



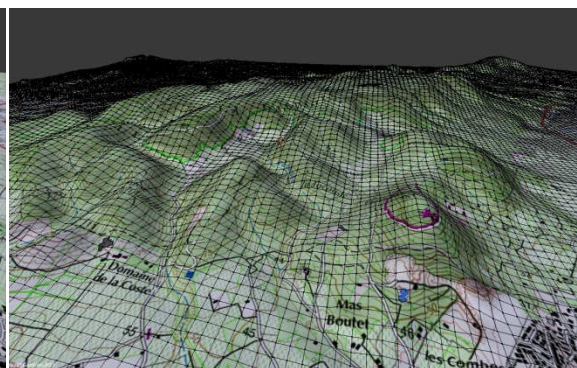
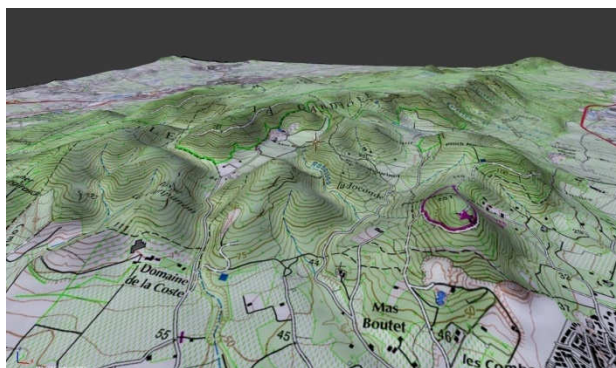
You need to know and specify the min and max elevation values (you can also do it later).

If you check *subdivide* the script subdivides the plane according to the number of DEM pixel which overlay the plane because the displacer needs some vertices to work with.

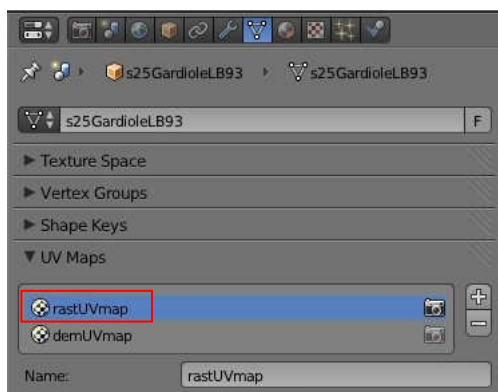
You can process a non scaled or a scaled DEM. Usually a DEM isn't scaled : in this example the elevation is range from 0 to 345m and the raster is coded into 16bits which range from 0 to 65535. With a scaled DEM the elevation value are stretched according to bit depth, in this case from 0 to 65535m.



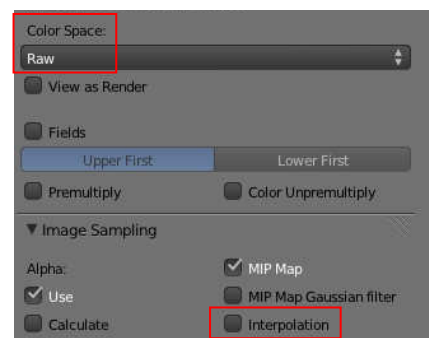
In Blender a non scaled 16 bit DEM appear fully black because the elevation values are very small compared to the highest 16 bit value (65535). With a scaled DEM we can see the elevation change.



Here the result, think to activate the smooth shader to get better visualization. You can play with the strength value to increase the warp effect, but you will lose the true elevation.



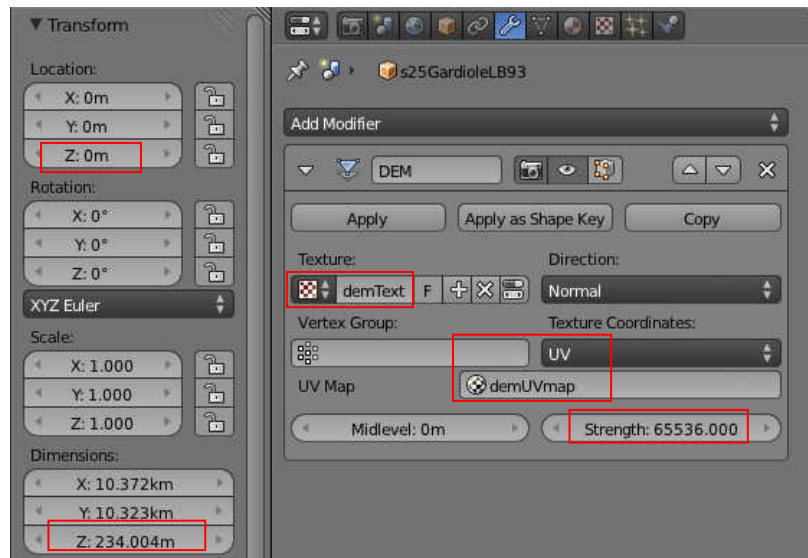
Note that the mesh as now a new UV map calculates according to the DEM overlay position



The script sets some important parameters of the displacer texture:

- Color space = Raw
- Interpolation = False

Here the modifier configuration:



- For convenience the mid level is always set to zero. This is not the min elevation.
- The strength value defines the vertex displacement:

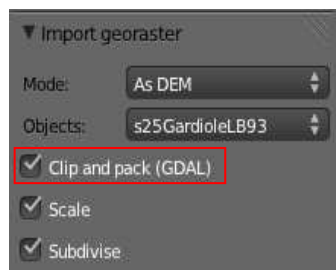
$$\text{Displacement} = (\text{Texture value} - \text{Midlevel}) \times \text{Strength}$$

In Blender color/luminosity values are between 0.0 and 1.0, so the strength will not be the same between a scaled and a non scaled DEM.

- With scaled DEM strength is the difference between min and max elevation
- With a non scaled DEM $\text{strength} = (\text{max} - \text{min}) / ((\text{max} - \text{min}) / 2^{\text{depth}})$
- The z location of the plane is set to minimum elevation, in this case to zero.
- You can also check the bounding box Z dimension, it must be equal to the maximum elevation value of the overlay part of the DEM. In this case the DEM is larger than my last imported raster so the Z dimension isn't necessarily equal to the DEM max elevation.

Note that the strength value is very high with a non scaled DEM. It's more convenience to use a scaled DEM with which the strength value is simply equal to the difference between min and max elevation.

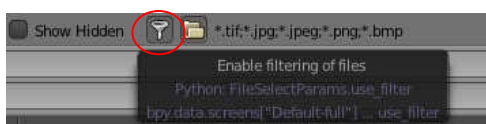
If you have [GDAL](#) installed (popular and powerful geospatial data processing library), you can process any GIS raster data format. The script automatically clip the DEM according to the working extent defines by the plane (so you can work with large data source) and pack the image into the *.blend file. It can also scale the DEM if you want.



You don't need to enter the min and max elevation, these values will be automatically computed.

The script always converts the DEM to 16 bits. So if your data is coded into 32 bits integer or floating values you will loss accuracy.

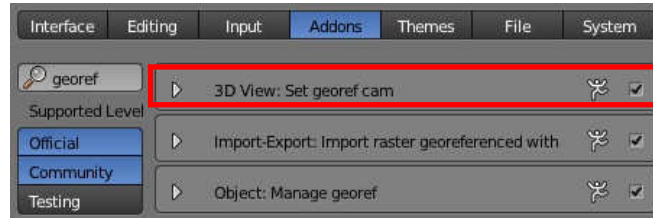
Note that the process creates temporary files (in Blender temp. folder) which will be removed later.



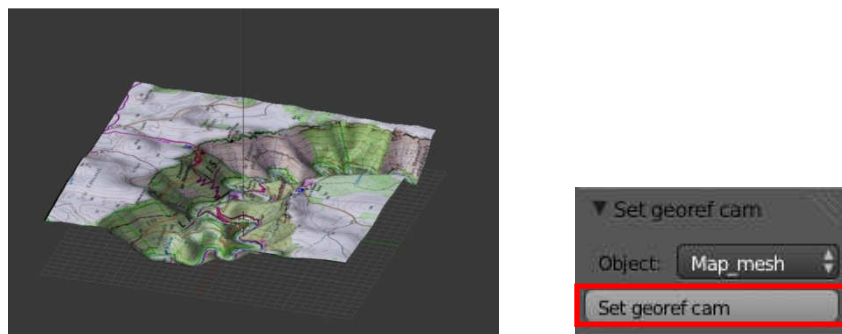
If you need to deal with GIS specific raster format and process it with GDAL, think to disable the filter.

Set up a camera for create a georeferenced rendering

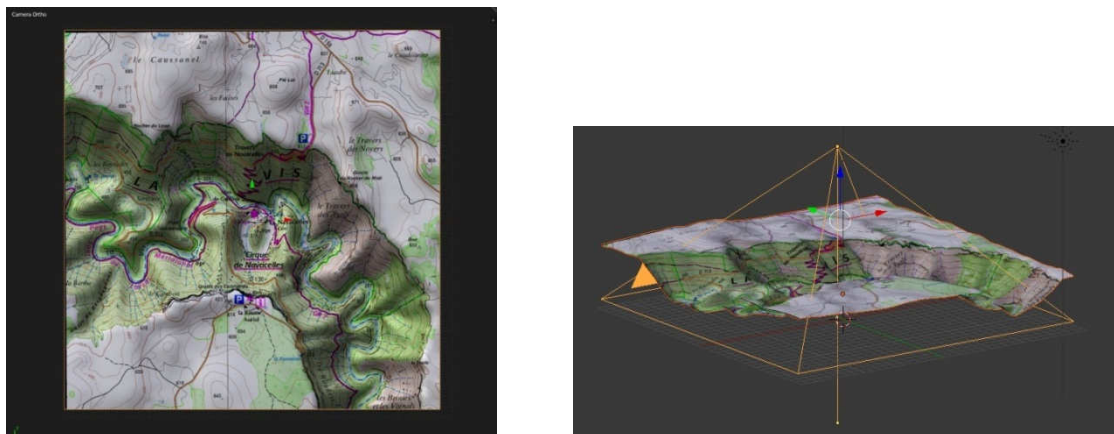
Enable the addon



If you have a georeferenced model, the panel will be active in the tool shelf

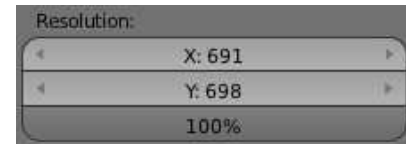
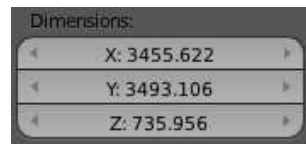
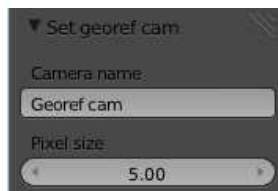


Choose the correct georef object that represents your map and then just click on the button to get a new camera with the good parameters.

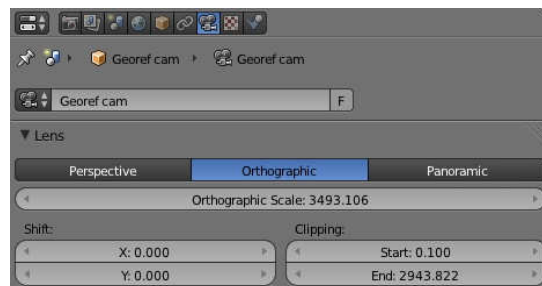


Tools options: you can name the camera and choose the target pixel size in map unit.

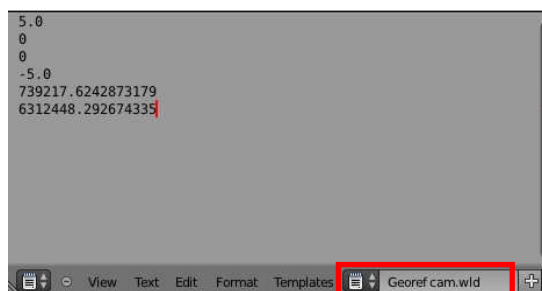
In this example the mesh dimension on x axis is 3455 meters, so if the target pixel size for the render is set to 5 meters / px, render resolution will be $3455 / 5 = 691$ px. **Set up the target size automatically sets the render size accordingly.**



The camera is set to ortho with the correct scale to enclose the mesh, its location is also set accordingly to the mesh position.



Now open a text editor and browse the loaded files, you will find the wordfile that contains the georeferencing data.



Now you can hit F12 and save the render output and the worldfile text on your disk. Don't forget to name them correctly and accordingly.

Open them in your GIS software:

