Index

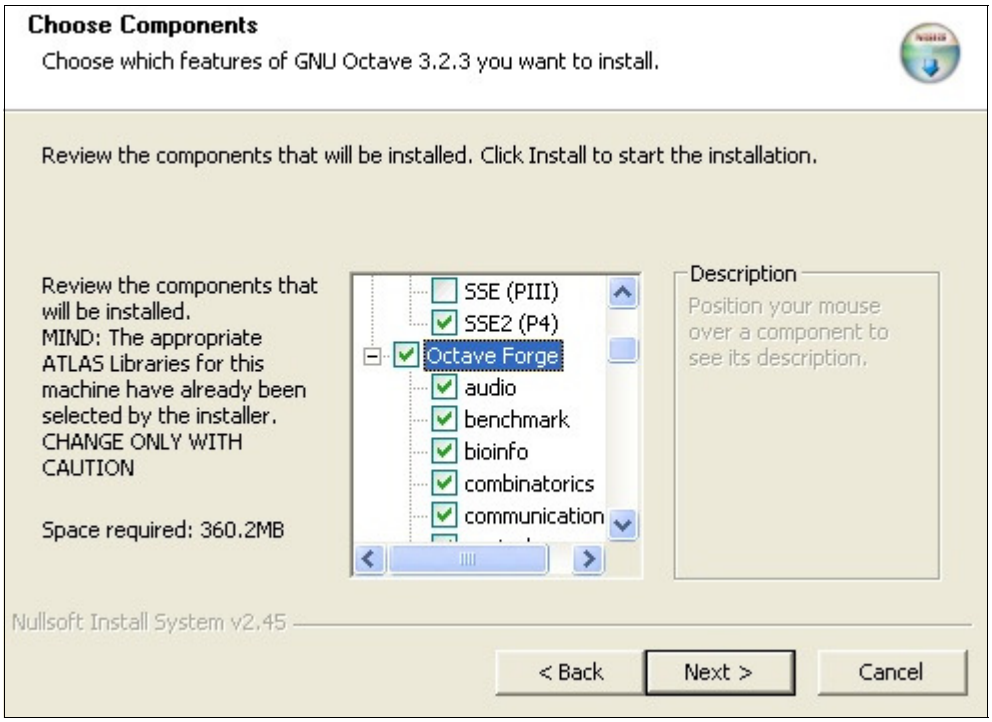
I. Software needed

I.1. Software list:

- Octave 3.2.3 (http://sourceforge.net/projects/octave/f...)
- MeshLab (http://meshlab.sourceforge.net)
- gmsh (http://www.geuz.org/gmsh/)
- 3D Route Builder (http://hybridgeotools.com/)
- BTBLofty (http://foro.simracing.es/attach...)

I.2. Octave 3.2.3 installation

- In the install step select Octave Forge libraries

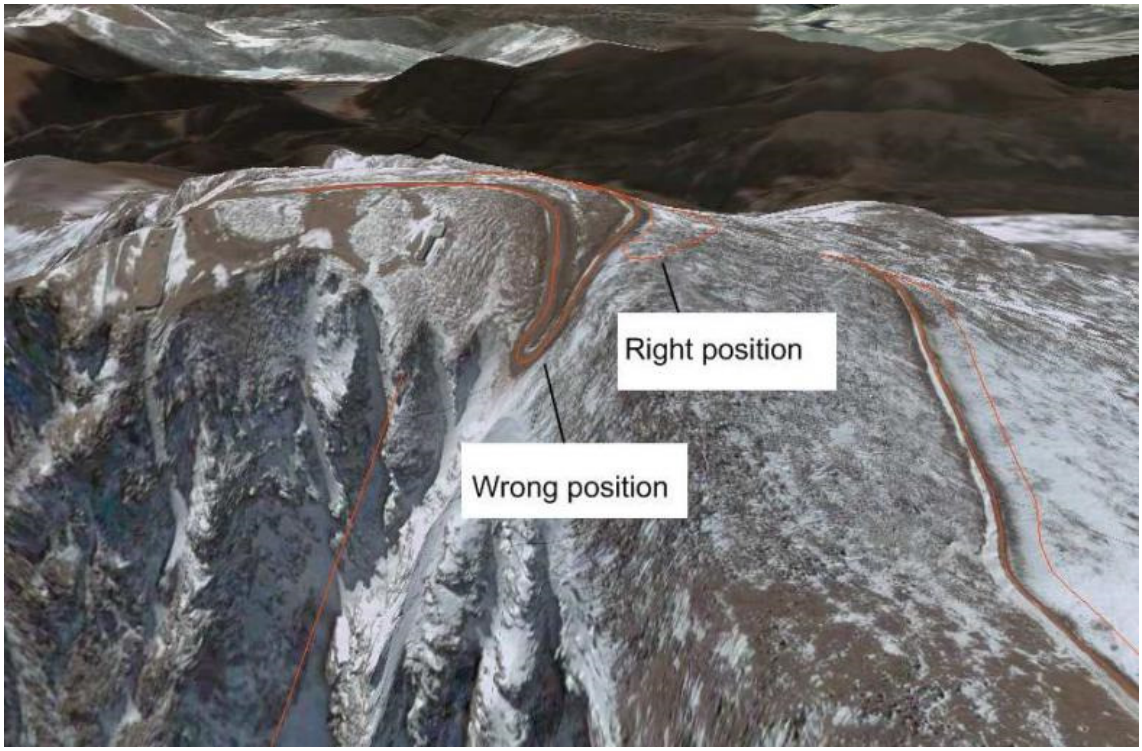


- **Alter installing Octave 3.2.3 run:**
  > pkg rebuild -auto communications
  Close Octave and start it again

II. Step 0: kml import

II.1. Importing the .kml file

The more precission we have in road's position, the better . Google Earth images can be useful for living the road a correct shape and width (and changes in width), but sometimes they are not useful for setting the right coordinates of the track. For example, the last hairpin of the Pikes Peak climb has, according to the GE's images, a huge slope (see image below). The true road position can't be that shown by the images. It is advisable to mark "road" layer in the left side of GE screen and use that information for making a better kml route. Images are a hint of true shape, but may be not real placement.



II.2. Number of coordinates in the .kml

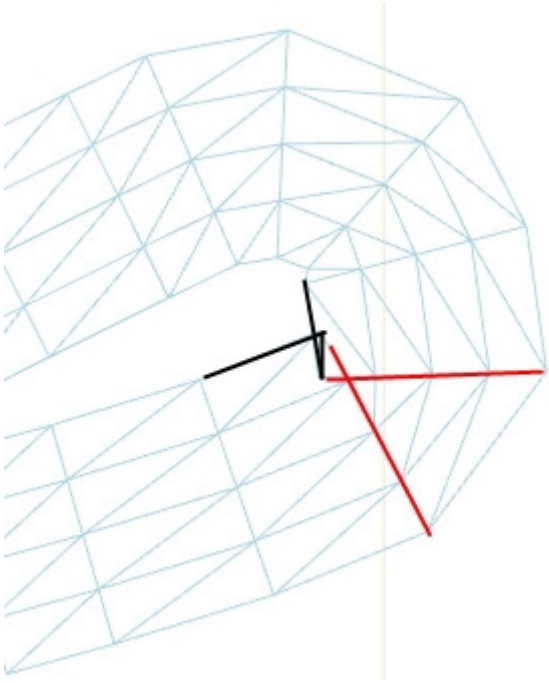
~~When using importakml script, the more points the .kml has, the better. If BTB0.6 is used for importing the .kml then may be fewer points are recommended.~~

Script *importakml* will use the points of the kml as spline nodes for the road, so you should chose them thinking how a smooth curve would join them.

II.3. Top view road debugging (only if BTB 0.6 has been used to import the kml)

If you use script importakml to import the kml file, the road will have soft turns and curves. This way it is unlikely to have problems in the meshing step. Nevertheless sometimes very closed curves or turns can make road polygons overlap causing the failure of the meshing process.

If we use BTB0.6 for importing the .kml, it is strongly recommended to debug the road in the top view, giving each curve the desired shape. Otherwise, if hairpins are not debugged very often the road overlaps itself. If this is the case, gmsh (the meshing program) will not be able to create a mesh). Knots in the road boundary must be avoided.



II.4. Venue.xml check (only if BTB 0.6 has been used to create the TerrainAnchors)

Once the terrain has been created (1 terrain panel with 0.1m width) we can check the result to detect potential problems. Just search for <Anchors (track anchors) and <TerrainAnchors (terrain anchors). The numbers on the right side of the “count” word must be even and identical. Were they not, the process can’t go on and we must have a look at the track, looking for overlapping zones.

Please note that the whole process is done with a single track segment. Splitting the track and the terrain in several pieces can be done later, at the final part of the process.

II.5. How can I manually create mapeo.txt? only if BTB 0.6 has been used to import the kml)

---

mapeo.txt establishes a correspondance between terrestrial and BTB coordinates. The info it contains are the coordinates of the same two points expressed in both coordinate systems. Knowing the coordinates for both points, it is possible to translate the coordinates of every point from one coordinate's system to the other.

Two points that are far away from each other are ok for this purpose. For example we can use the coordinates of the first and last points of the road, if they have very different longitude and latitude coordinates. From the .kml we would copy the first and last couple of coordinates, and from the Venue.xml we would copy "x" and "z" values of the first and last nodes of the imported road.

Example mapeo.txt:
2157.232
5137.749
-105.037205228836
38.9209304085051
1666.922
-3785.737
-105.042881979876
38.8405926230566

Reading:

2157.232 -> x of the first node
5137.749 -> z of the first node
-105.037205228836 -> first kml point longitude
38.9209304085051 -> first kml point latitude
1666.922 -> x of the last node
-3785.737 -> z of the last node
-105.042881979876 -> last kml point longitude
38.8405926230566 -> last kml point latitude

Please note that if mapeo.txt is created manually, the x and z (or longitude and latitude) coordinates of the two points selected must be quite different. If the road starts and finishes on points that have similar x or z values, you can't use starting and finishing points for creating mapeo.txt.

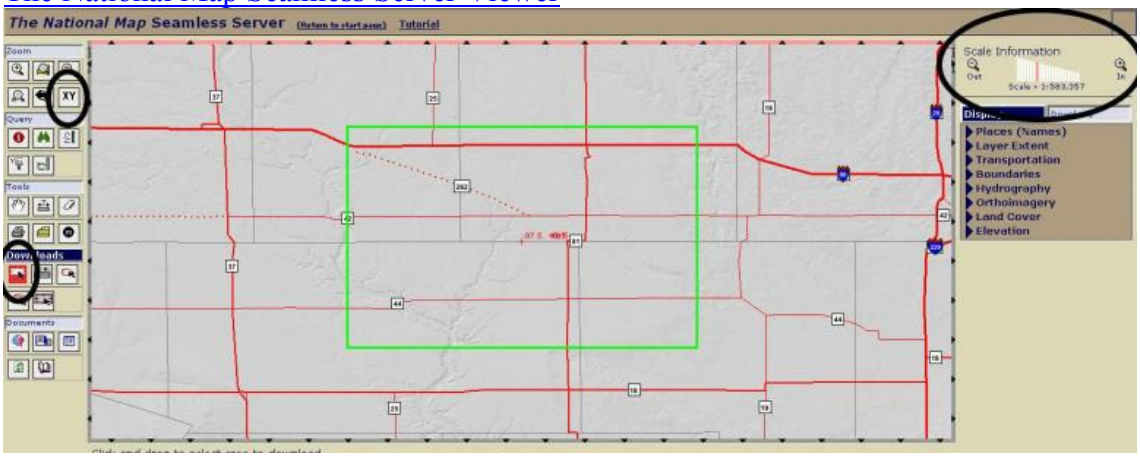
If mapeo.txt is created by hand, it is recommended to check that the coordinates of the points lead to the same place in GE and BTB. We should run BTB and watch the position of the first point (x and z) and then run GE and check that the terrestrial coordinates lead to the same place. The checking process should be repeated with the second point..

III. Step 1: Getting elevation data

I.1. Seamless server

This web offers for free elevation data with 10 or 30m resolution for USA and 90m (and often with missing point values) for the rest of the World.

The National Map Seamless Server Viewer



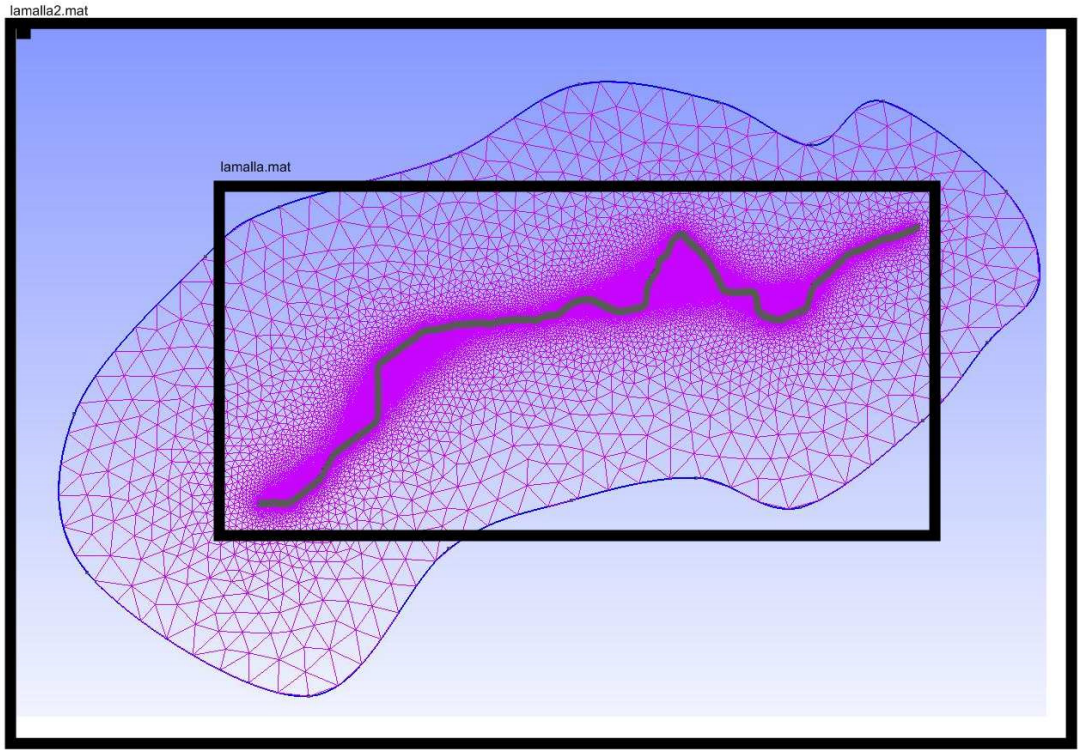
I. To check what kind of data they have from our zone of interest we select the tool **Zoom->XY** and write the coordinates of a point. Then we zoom in (upper right side of the screen). With **Downloads->*Rectangle*** we frame the interesting zone. If the rectangle is too big, it will turn red. In the pop-up window we click on **Modify Data Request** button. We look for the **Elevation** section and we select the data source,. For example "**National Elevation Dataset (NED) 1/3 Arc Second**" and we choose "**GRIDFLOAT**" format. We click on "**Save Changes & Return to Summary**" button and if no problema arises a Windows with a "**Download**" button will pop up. This is the easiest an quickiest way to get good elevattion data. If we are not interested in USA tracks we can always use GE through 3DRoute Builder or BTBLofty.

   Note:
   3 arcosecond ~= 90m
   1 arcosecond ~= 30m
   1/3 arcosecond ~=10m

I.2. Using 2 grids

We want good elevation data resolution near the road but **it is useless having 10m resolution 1Km away from the road** because we will create a mesh with big (75-250m) triangles far from the road. We can create two grids with elevation data: one with good resolution at least covering the road and one with worse resolution covering the whole terrain of the project.

For the bigger grid we can use a second folder called s2_elevation_b. The elevation data will be automatically copied to s4_terrain folder as file **lamalla2.mat**. The name for the smaller grid file is lamalla.mat ("la malla" means "the grid" in spanish language).



I.3. Terrain layer in Google Earth

If "terrain" layer isn't checked (lower left zone of the GE window), 3DRB and BTBLofty will return 0m elevation value for all the points of the .kml.

I.4. Data retrieved has errors, how can I correct them?

If you run read_grid and you see that some of the values are not correct, you can open the gridXXX_relleno.kml files with Google Earth and detect which of them has the mistakes. Redo the process with 3DRB or BTBLofty and check again.

II. Step 2: Using Terrain Anchors

The following description is about TerrainAnchors generation using BTB 0.6. Script btb06 makes use of BTB 0.6 not mandatory. Ignore this text if you are not using BTB 0.6

Why do we create terrain with a width of 10cm?

Terrain and road are linked at points called (road)Anchors. The exact position of the Anchors are not explicit in the Venue.xml. Creating a terrain with small width (10cm) we create terrain nodes (called TerrainAnchors) that are very near of the (road)Anchors, just 10cm away and their position do appear explicitly in the Venue.xml. If everything runs ok there will be the same amount of Anchors and TerrainAnchors and we could use the TerrainAnchors position to create a mesh for the terrain linked to the road.

If we have created, i.e., 7870 TerrainAnchors, the first half, from 0 to 3934 give as the position of the Anchor son the left side of the road, while the rest, from 3935 to 7869 are also Anchors that cover all the road from the start to the end, but on the right side.

How can I create the different .xml files needed in venue folder from Venue.xml?

- All the text before <*Terrain*> (<Terrain> also) should be saved as **pre_terrain.xml**
- All the text after <*/Terrain*> (</Terrain> also) should be saved as **post_terrain.xml**
- Terrain anchors, from <*TerrainAnchors* to <*/TerrainAnchors*> shpuld be saved as **anchors_s1.xml**. Terrain anchors are the part of Venue.xml starting after <Terrain>

- Road nodes (or track nodes) are the text between <*nodes* and <*/nodes*> (both included) and should be saved as a file called "**nodes.xml**":

- The definition of road Anchors as a percentage distance between nodes are the text between <*Anchors* and <*/Anchors*> (both included), and should be saved in a file called "**porcentajes.xml**"

III. Step 3: Raising the road

III.1. What is the function of dar_altura parameters?

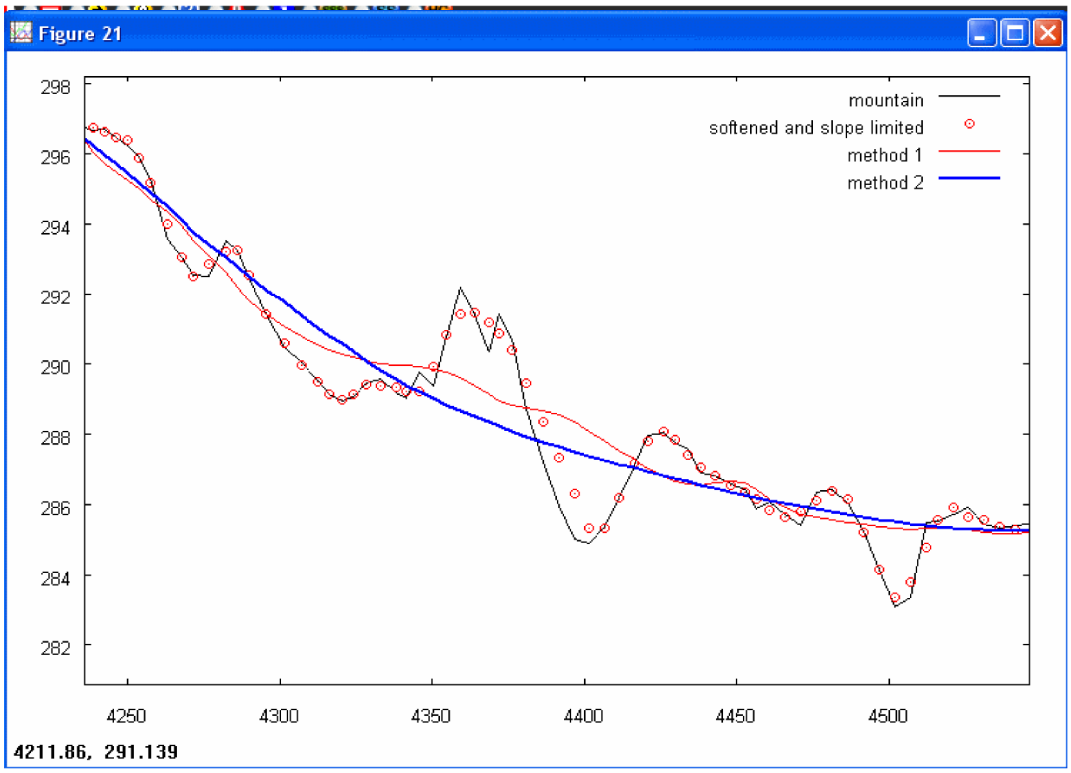
Parameters are a smoothing factor and maximum and minimum slopes allowed.

script ***creartrack1*** gets the height of the road according to a) the terrain elevation data and b) the position of the Anchors. Script **dar_altura** shows creartrack1 output with a blue line. In first place *dar_altura* modifies the elevation of the road using the slopes' constraints and the result is show on the graph with red circles. Finally elevation values are smoothed using the smooth factor specified. The smooth factor must be always odd. If, for example, we use a smoothing factor of 25, each point of the road (each Anchor) will be given a elevation value that is the mean value for 25 points centered on that point. The distance taken into account when smoothing depends on the anchor's separation. The default anchor separation has a mean value of 5m.

The black line is the elevation that the road should have according to the terrain elevation data available (lamalla.mat).

Red circles try to fit the blue line, but without exceeding maximum and minimum slope constraints.

The **red line** is the elevation that the road will have, and is the result of smoothing the position of the read circles using the smoothing parameter desired. If you want to use this values, named method 1, for the road, add a 0 as 4th parameter to the call of dar_altura.
The **blue line** is the elevation that the road will have, and is the result of fitting the red line with a smooth curve (splines). A 4th parameter can be added to dar_altura if you want to change the default spacing of points used for defining the splines. Default is 25m. The bigger, the smoother.



III.2. I want to test several posibilities for the parameters of dar_altura, do I have to run creartrack1 each time?

No. That is the reason why there are two separated scripts. Once you have run *creartrack1*, you can run *dar_altura* several times with different input parameters. Only if anchor's position or lamella.mat are changedit is mandatory to run again *coge_datos* and *creartrack1*.

III.3. How can I edit the elevation profile created by dar_altura?

(There is a javascript tutorial inside documentation\tutorial_dar_altura.7z)

Run dar_altura as usual

> dar_altura(15,0.3,-0.3)

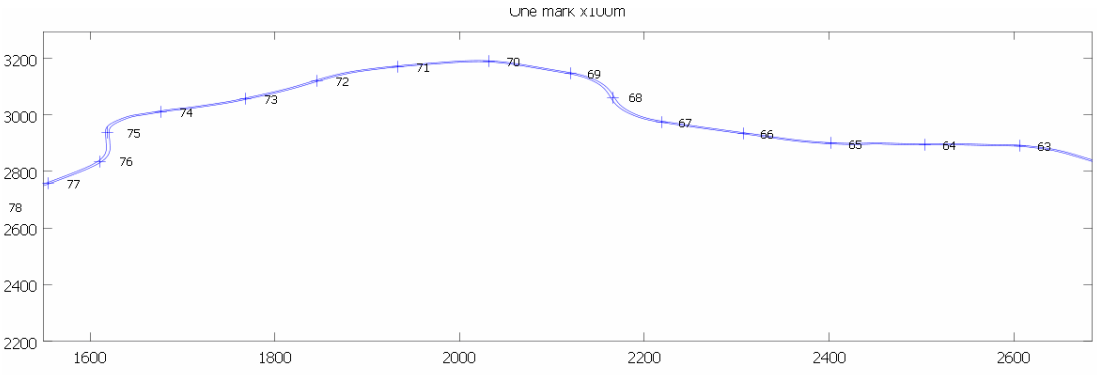
```
> dar_altura(15,0.3,-0.3)
Leyendo ..\anchors.mat
Leyendo alturas_track1.mat
Leyendo el fichero retoques.txt
Cerrando el fichero
        e-_ end             n-_ new segment
```

Now **dar_altura** waits for the user to press "n" or "e" and <ENTER>

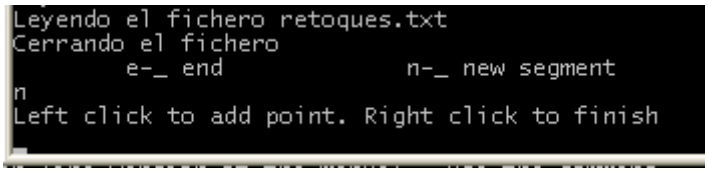
- "n" if you want to change a segment of the elevation profile

- “e” if you want to quit

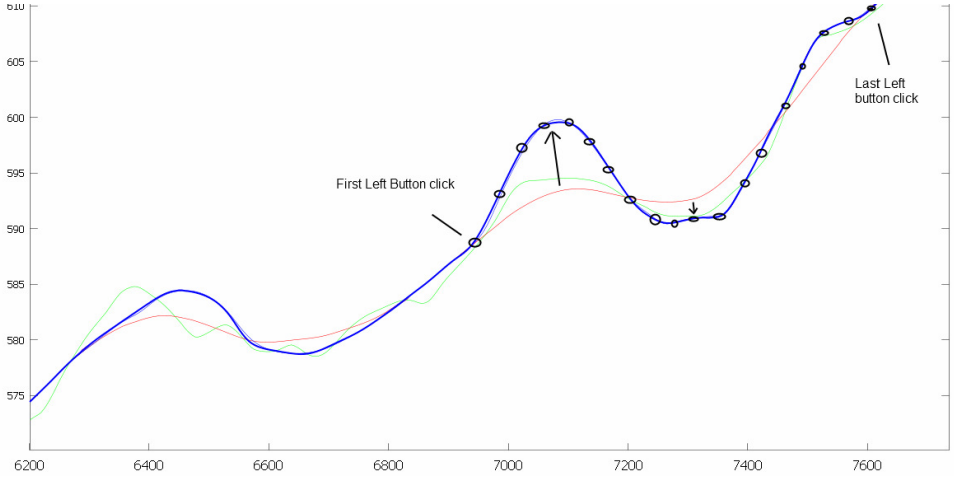
But **before choosing “n”, zoom the zone you are interested on**. Before zooming you probably will need to have a look at the other figure created by dar_altura. It shows the distance travelled at each point of the road (scale by 100 to get distance in meters)



If for example you want to change the elevation profile between position 6800m and 7600m, then you know you need to zoom that zone (select zone with right button) on the other figure. After zooming you can press “n” on the textmode window. Message on textmode window should change:



Now you have to click on the elevation profile graph using the left button of the mouse. You can use any number of points (>=3). When finished, click anywhere with the right button.



Then you will be asked again to choose “e” or “n”.

Once you press “e” the script dar_altura exits. Inside “s3_road\salida” you can find a file called “Venue.xml” that can be opened with BTB to check the shape of the road.

Important: All changes applied are recorded in a file called "retoques.txt". This is a plain-text file that can be edited by hand if needed. Each line of the file is composed by the number of points of a segment and the (x,y) coordinates of those points.

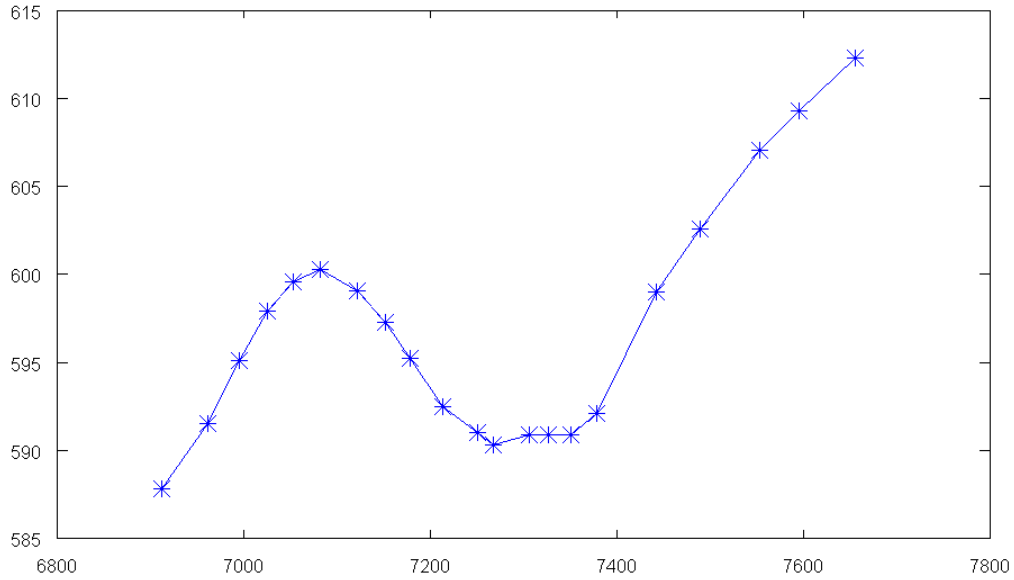
Each time dar_altura is called, 'retoques.txt' is read and elevation profile changes are applied. New changes will be added at the end of 'retoques.txt'.

If you want to remove old changes, delete "retoques.txt" file. Lines inside retoques.txt are applied in the same order they appear inside the file.

Example retoques.txt line:

21 6911.3 587.8 6961.6 591.5 6994.3 595.1 7024.5 597.9 7052.2 599.6 7081.2 600.3 7121.5 599.1 7151.7 597.3 7178.1 595.2 7213.4 592.5 7249.9 591.0 7267.5 590.3 7305.2 590.9 7326.6 590.9 7350.6 590.9 7378.2 592.1 7442.4 599.0 7489.0 602.6 7553.2 607.1 7594.7 609.3 7655.2 612.3

If we plot those points we see they are the definition of one of the changes applied:

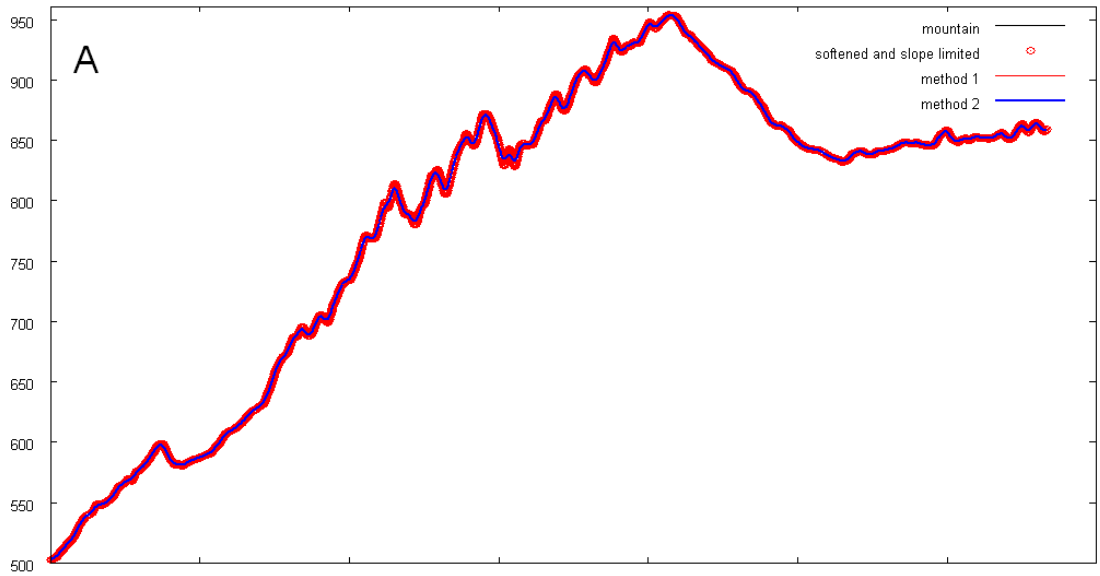


The first parameter of dar_altura is a smoothing factor. You must choose it so little manual changes are needed. It must be always odd.

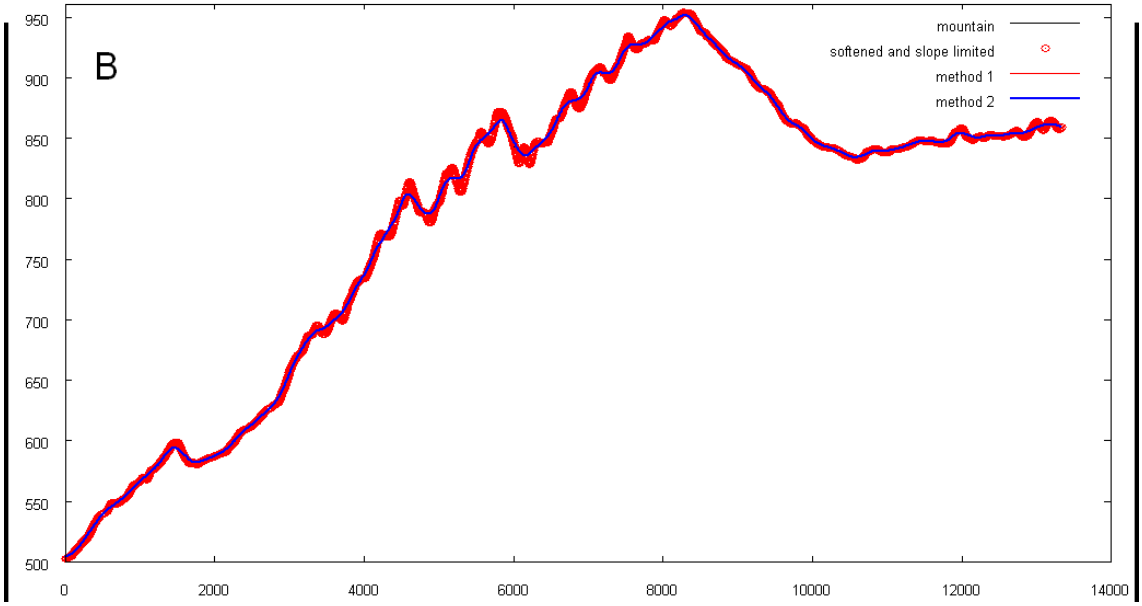
The final elevation profile will be a spline created using one elevation value each 25m. That means that *little irregularities created by manual editing may be NOT so important*. If you want you can change that distance (25m by default) setting it as the 4th parameter of dar_altura.

III.4. Usage of script "corregir"

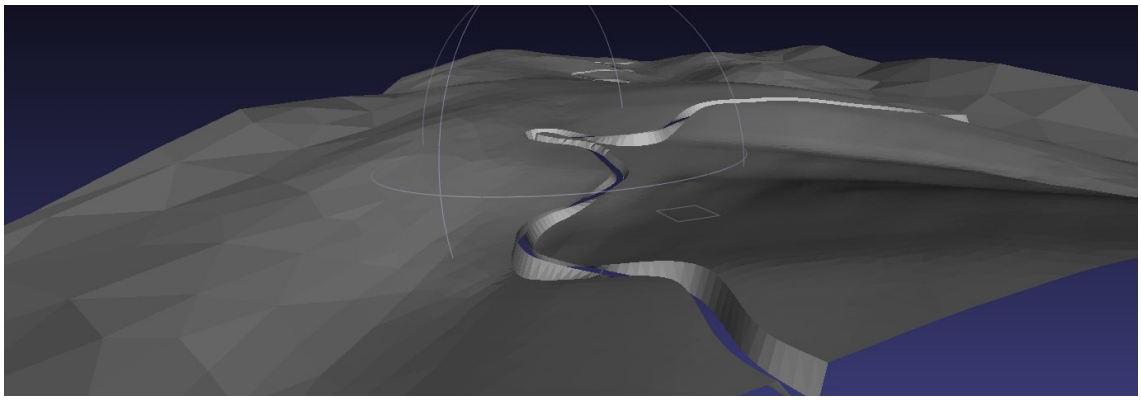
Let's assume we want to create a track but when we run "dar_altura" we detect elevation data is no good, and the road goes up and down in a nonsense way (see Fig A). Elevation changes are obviously caused by bad elevation data.



Script “corregir”, can change elevation data in the following way. First we create a softer road with dar_altura (see Fig B).
> dar_altura(53,1,-1,100)

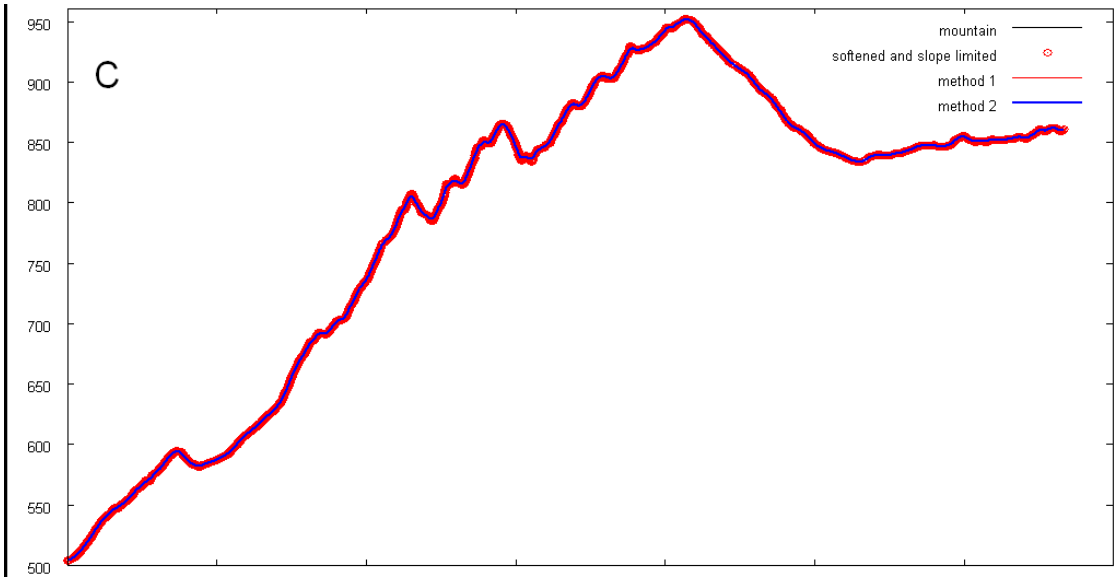


This road does not fit the mountain well. If we went on with the process using this road, we could get something like this:

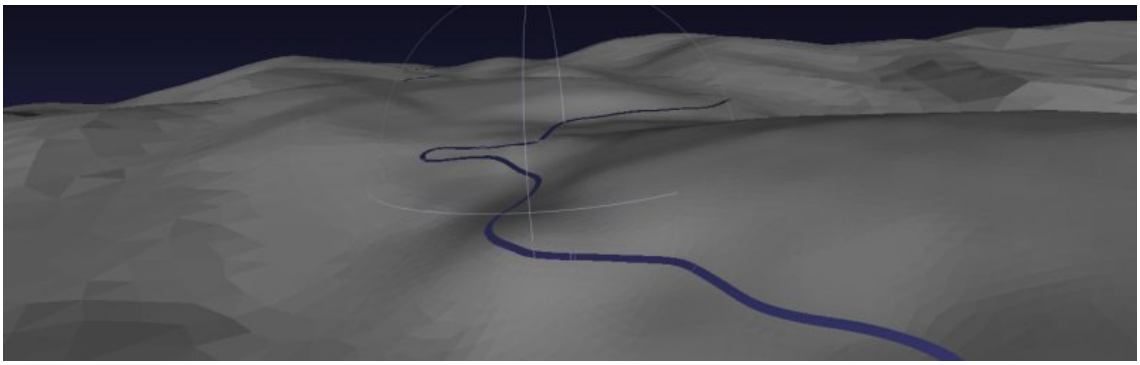


So, now we want to change the elevation data to fit our desired “soft road”:
> corregir
And we run again dar_altura to create the final road, a road that fits the new elevation data:
> dar_altura(21,0.25,-0.25)

We can see (Fig C) that now the road (blue line) is almost as soft as the desired one, and the mountain (red circles) hasn’t big elevation differences with the road. Compare differences between red circles and the blue line in Fig B and Fig C. The road is almost the same, **but “corregir” has changed elevation data to fit the road**.



Once we have a road with a elevation profile we like, we go on with the next script. The result can be as good as:



Scripts we have run (you can use the parameters you want for them):

```
cd ..\s3_road
coge_datos
creartrack1
dar_altura(53,1,-1,100)
corregir
dar_altura(21,0.25,-0.25)
```

If you want to start the process from scratch, just run again coge_datos inside s3_road, and you have the original elevation data ready for creartrack1 and the rest of the scripts. If you don't want to use "corregir", just use dar_altura as usual.

IV. Step 4: creating a mesh for the terrain

IV.1. What is anchors_carretera.geo?

Anchors_carretera.geo is a file that should be processed with gmsh. It contains:

- TerrainAnchors, located 0.1m away from the real road Anchors. They are joined with straight lines.
- Help points, on both sides of the road at a distance specified as a parameter.
- Already created “Plane surfaces” for the driveable zone.

IV.2. gmsh basic concepts

If two points of the boundary of a mesh are joined with a straight line, both points will be part of the mesh.

If two points of the boundary of a mesh are joined with a spline, those points will not necessarily be part of the resulting mesh. gsmh will find the optimal position of the triangle nodes on that boundary.

When working with gmsh each point has a "**characteristic length**", that is the lenght we want for the triangle sides on this point. For example, inside the file anchors_carretera.geo (it is a plain text file), the road Anchors haver a characteristic length called "cl1", with a value of 20m. Nevertheless Anchor points are joined with straight lines, so triangles in touch with the road will have a side length defined by the position of the Anchors (mean of 5m).

IV.3. gmsh basic controls

- Right mouse button and drag: move the project
- Ctrl + left Mouse button and drag: make zoom of a zone
- Click 1:1 text on the left-down corner: Fitting zoom
- Click Z on the left-down corner: restore top view
- Left Mouse button and drag: 3D rotate view
- Mouse wheel: zoom

IV.4. What are the limits for the non-driveable zone?

The non-driveable zone shouldn’t exceed available elevation data. Otherwise scripts like procesar_nodostxt will fail. If you want to see the limits on the gmsh screen, run addgrid inside s1_mesh folder:

> **addgrid(1,1)**

Where the ~~first~~ 2 parameters are the number of horizontal and vertical divisions of the grid ~~and the last parameter sets the start number for the points of the grid. The last value should be high enough to avoid using an already existing value.~~

Do NOT reach the limits with you gmsh meshes. Otherwise, next sripts will fail as there will be no available elevation data for some points.

IV.5. How should I create the external Boundary of the non-driveable zone?

Before meshing we should create the "Plane surfaces" to mesh. To create the plane surfaces we must have boundaries defined, and the definition of boundaries need points to be defined.

We select:

Geometry->Elementary Entities->Add->New->Point

Once the Mouse is on the desired position for the point we hold "*shift*" on the keyboard, and the coordinates get frozen. We move the mouse to the "Contextual Geometry Definitions" window and there we set coordinate Z=0 and Characteristic Length to *cl3* (cl3 is a value defined at the beginning of anchors_carretera.geo). Click Add.

Once we have defined all the points we need, we select:

Geometry-> Spline

and we define the spline shape selecting the points the spline must follow. Once the curve has been close we can define a "Plane surface" defined by the spline as its external Boundary and the driveable surface as its internal hole.

Geometry -> Plane Surface

We click on the external Boundary. If it has been correctly closed we will be asked to select the hole boundaries ("Select hole boundaries"). We must select all the segments of the driveable zone boundaries, until all the segments are colored red. At that moment the option "undo" will disappear from the top part of the window. Then we press "e" to end the definition of the "Plane surface".

We can then mesh to check that everything is ok:

Mesh -> 2D

and all the already defined plane surfaces will be meshed.

We need at least two kinds of surfaces:

1. Driveable: help points "minus" road

2. Non driveable: external Boundary “minus” help points.

We can select Tools->Statistics to know how many triangles will have our mesh.

IV.6. What is the output of the process?

If we save the mesh a file called anchors_carretera.msh will be created. This file contains two kinds of lines:

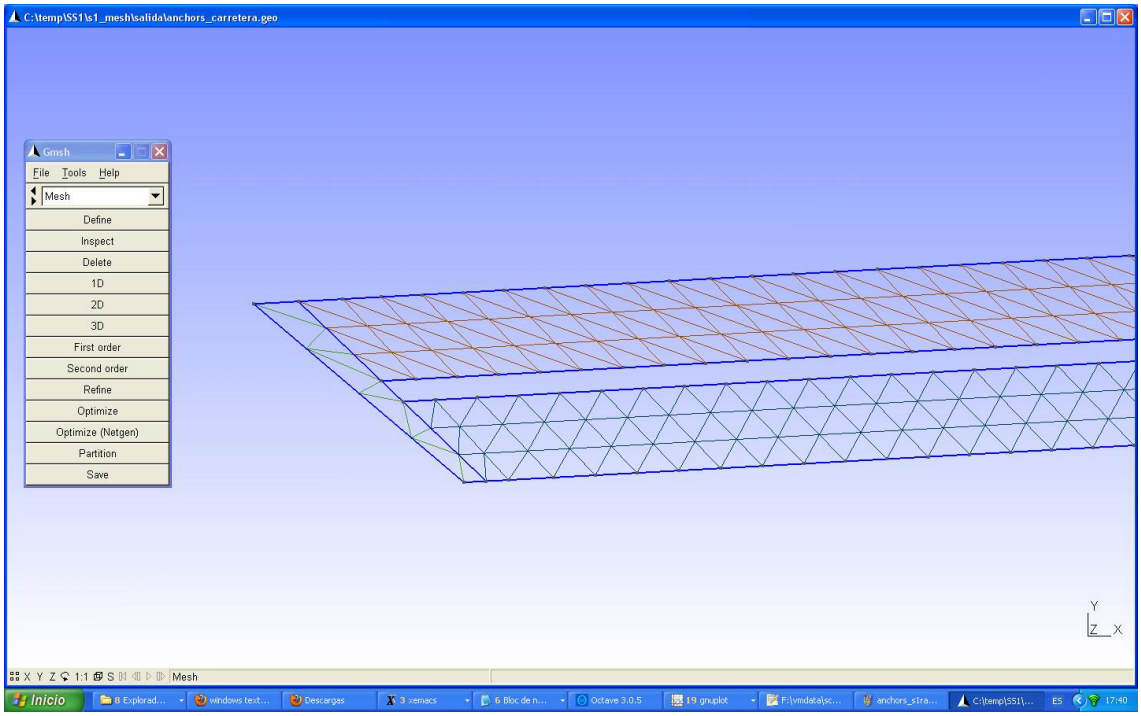
1) Nodes. For example node #14:

14 2149.778 5113.46 0

2) Triangles (gmsh calles them elements, and they are called faces inside BTB). For example triangle 21222 belongs to the physical surface of driveable ones (111) and is defined by three nodes: 1490, 17130 and 1491:

21222 2 3 111 31672 0 1490 17130 1491

IV.7. How are the plane surfaces that I must add on both ends of the road?



IV.8. How can I define physical surfaces?

There are 2 physical surfaces, driveable (numbered 111) and non-driveable (numbered 222). Each one of them is defined as the list of "plane surfaces" that belong to that physical surface.

~~Physical Surface 111 has already been defined inside anchors_carretera.geo~~. The initial list of surfaces for Physical Surface 111 is the contents of file salida\phys111.txt (it is this way since multitrack support was added. Read *Multi track scripts.doc* for more info). If your project has no additional tracks, just add the contents of phys111.txt at the end of anchors_carretera.geo: "**Physical Surface(111)= {contents of phys111.txt};**".

But is is not completed. We must add, at least, the code of the two plane surfaces that we have created on both ends of the road. If you have followed the order exposed in the quick guide, just search backwards form the end of the file the text "Plane Surface" and the second and third commands found will be those that define those plane surfaces).

We add "**Physical Surface(222)={X};**" at the end of the file, where X is the code of the last Plane Surface defined inside anchors_carretera.geo (it is the non-driveable plane surface).

Only the plane surfaces that belong to a physical surface will be part of the .msh file, so it is important to pay attention when defining the physical surfaces. **Once you have finished meshing it is recommended to open anchors_carretera.msh with gmsh to check that all the meshes are there**.

IV.9. Can I add random elevation changes to the terrain?

There are 2 ways:

procesar_nodostxt admits as parameter the maximum elevation noise (in meters) desired. If, i.e., we run **procesar_nodostxt(0.5)** the elevation of each node is increased a random value between 0 and 0.5m. It also admits lower and upper noise limits:
> cd ..\s4_terrain
> **procesar_nodostxt([-0.5 0.5])**

But all the changes we make with procesar_nodostxt affect the way the meshes are simplified, so may be a better way is using **terrain_noise** script. Once we run juntar_mallas and any time before running join_all we can add random variation to the nodes of the mesh:
> cd ..\s4_terrain
> **terrain_noise([-0.25 0.25])**

IV.10. How do I use MeshLab to simplify the terrain mesh?

Inside octave (s4_terrain folder) we run

> **simplificar**

and the mesh is splitted in three parts that will be processed separately:

1. Triangles linked to the road We will not touch them because we want small triangles in contact with the road. File **intocables**.ply
2. The rest of driveable triangles. File **conducibles**.ply
3. Non-driveable triangles. **noconducibles**.ply

The output from *simplificar* are three meshes with a format called ply. These files will be opened and processed with MeshLab:

1) **intocables.ply**

Filters -> Cleaning and repairing -> Remove unreferenced vertex

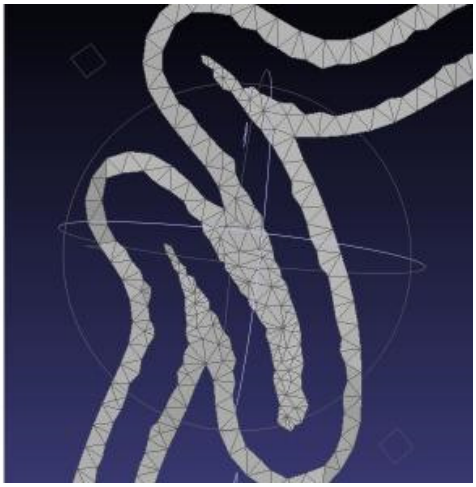
Save the result as **i.ply** (don't save it in binary format)

2) **conducibles.ply**

Usually there is no need to simplify this triangles so

Filters -> Cleaning and repairing -> Remove unreferenced vertex

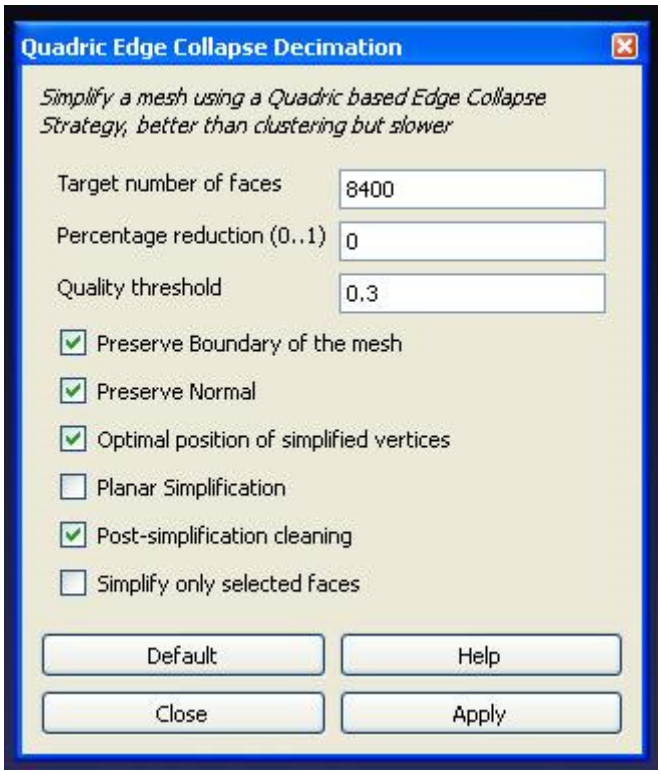
should be used. But if there are zones (inside hairpins, see following image) that have too many triangles you can use



Filters-> Remeshing, simplification and reconstruction -> Quadratic Edge Collapse Decimation

to simplify those parts.

It is very important (mandatory) to select the option "**preserve boundaries**". Other interesting options to select could be optimum positioning of simplified vertex and post-simplification cleaning. It is encouraged to check that the resulting mesh is ok and that there are not strange artifacts that sometimes show up.



The result must be saved as **c.ply** inside the "salida" folder (don't save it in binary format).

3) noconducibles.ply

Filters-> Remeshing, simplification and reconstruction -> Quadratic Edge Collapse Decimation

Again, it is very important (mandatory) to select the option "**preserve boundaries**". Other interesting options to select could be optimum positioning of simplified vertex and post-simplification cleaning. It is encouraged to check that the resulting mesh is ok and that there are not strange artifacts that sometimes show up.

Save the result as **n.ply** inside "salida" folder (don't save it in binary format)

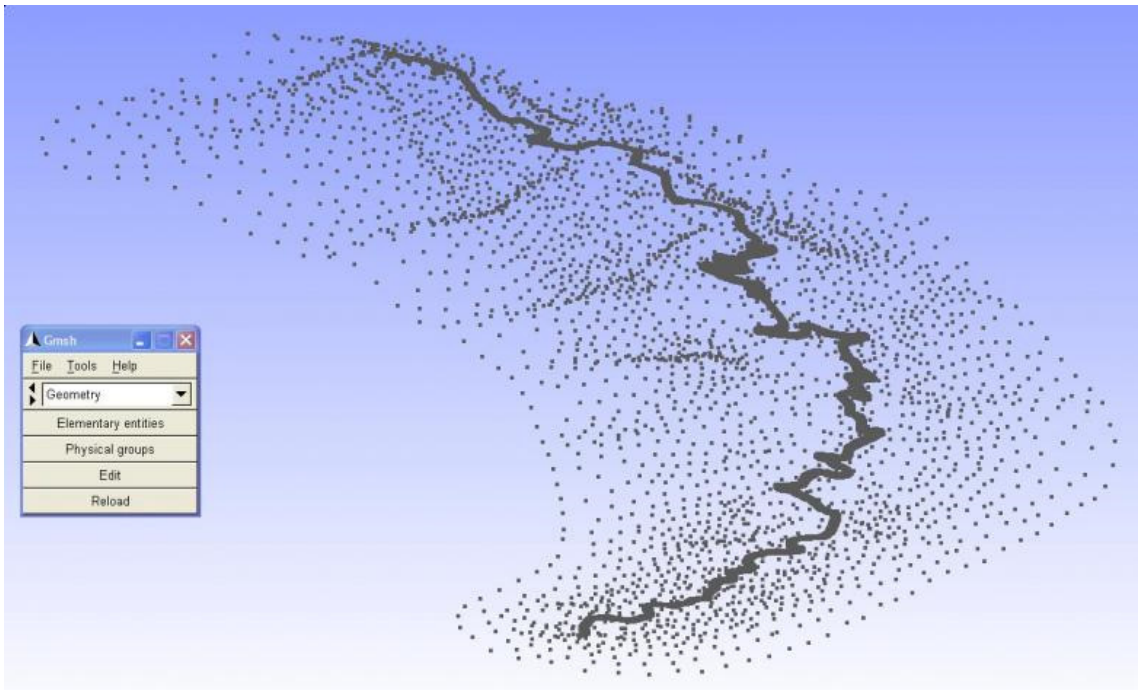
Note: sometimes MeshLab says there is an error when opening a .ply file. If that were the case, the error message disappears if "short" word is changed to "int" in the property line of the file (9th line).

Now we join the ply files:

> **juntar_mallas**

The output of this script are a new list of triangles (file elements.txt) and a new list of nodes (file nodos_con_altura.txt) for the joined mesh.

File salida\prueba.geo can be used to view the nodes of the mesh.



V. Step 5: BTB terrain generation

V.1. What is the format of a terrain triangle inside BTB?

Creating a list of triangles in the format hended by BTB is easy. It is as simple as writing down the number of the three nodes that define the triangle. Nevertheless in BTB the texture of a triangle of the terrain can be a mixture of two textures. In this case for each triangle we must specify the degree of mix (0->all primary, 1-> all secondary texture) and each node. The scripts assume for the sake of simplicity that every triangle belongs to a zone where there is a mix of textures. The degree of mixture is the value pf the P parameter inside the TerrainFace:

```
<TerrainFace Selected="False">
  <Anchor0 Props="A4825" P="0.00" />
  <Anchor1 Props="A11644" P="0.44" />
  <Anchor2 Props="A10839" P="0.45" />
</TerrainFace>
```

On the other side terrain anchors are named as **AXXX**, as shown in the image above, but road Anchors are named depending on the track segment they belong to. For example road Anchors from the first track segment will be named **T0 XXX** (T0 for the first segment, T1 for the second, etc.).

Example:

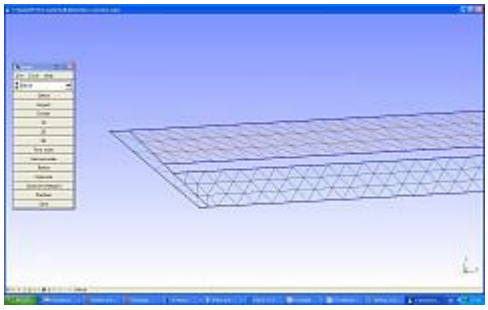
```
<TerrainFace Selected="False">
  <Anchor0 Props="T0 4825" P="0.00" />
  <Anchor1 Props="A11644" P="0.44" />
  <Anchor2 Props="A10839" P="0.45" />
</TerrainFace>
```

It is posible and easy to change *procesar_elements* or *procesar_elements_mt* to give each zone of the terrain a different texure mix degree depending on their distance to the road or their absolute elevation.

VI. Step 5: The invisible protection wall

VI.1. I can't get the walls created. The process stops

The walls are created in the interface between driveable and non-driveable parts. In order to make the scripts do their job it is mandatory that in every part of the track there are triangle on both sides of that interface. The most problematic zone are the ends of the road because plane surfaces are not created automatically there and the user has to create them and include them in the list of plane surfaces that belong to Physical Surface 111.



VII. Step 8: Terrain and road splitting

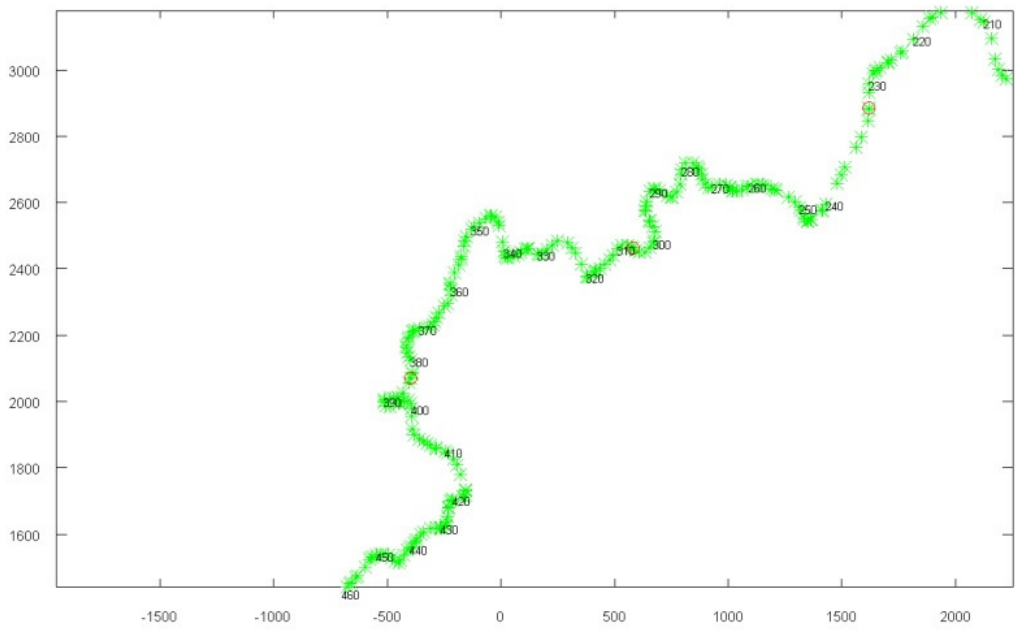
VII.1. After splitting my road has a wrong width

The list of tracks created by *partir_track* are build with a default profile, material and the road width used as a parameter for mallado_regular. If you want to change that value just open the Venue.xml and replace
<WidthMultiplier>Y</WidthMultiplier>
with
<WidthMultiplier>X</WidthMultiplier>
where X is the scaling factor for our road (usually 0.5 for a 5m width road). If you have 8 track segments, you should replace the text 8 times.

VII.2. Can I change the points where the road is splitted?

Yes. For this purpose we must first run **split_track(n)**, being n the number of segments we want to create.

split_track creates a file called '**pos_nodes.txt**" that contatins the numbering of the nodes where splitting will take place. split_track also shows a graph so we can check those points on the track top view. If we want to change the split positions we just have to **edit pos_nodes.txt** and run partir_track. partir_track uses pos_nodes.txt as input so it determines the number of segments (and where they start and finish).



VIII. More scripts: adding crossing tracks

Is there an easy way of adding another track?

This is obsolete. May be you should read Multi track scripts.doc

Let's assume we want to add a crossing road, or a bifurcation and we want that track to be on its exact position. Ok, then we get the kml of that track, we remove the initial and final parts, leaving only the coordinates (longitude, latitude,elevation) and we save it as crossing01.txt. Then we run:

> cd f:\project\s4_terrain
> **convertir('crossing01.txt','nodes01.txt')**

The output file, nodes01.txt will contain the list of nodes for that track. With that info we can create a track inside BTB in the right position. If we are using BTB 0.6 we could create a track incide BTB with a couple of nodes (not important where they are), save the project and replace the listo of nodes of that track with nodes01.txt

This script could be useful also for getting the coordinates of an object in the project knowing its terrestrial coordinates.

Care should be taken when inserting tracks because BTB will take them by default as ground for 3D objects and existing 3d objects may get raised accidentally. Please note that the tracks created are not smooth in top view nor in elevation.

IX. More scripts. Pacenotes generation (changing pacenotes.ini)

This process is not specific for tracks created with the scripts. It could be useful for every track exported to be used with RBR_RX plug-in, even if it has nothing to do with the scripts.

Extract the folders structure, for example in a folder **c:\temp\track**
Copy the files **pacenotes.ini** and **driveline.ini** to c:\temp\track\pacenotes folder

5) Run octave and type:

> cd c:\temp\track\pacenotes

> addpath('c:\temp\track\scripts')

> pacenotes

> pacenotes2(0.01,10)

You will get a new **pacenotes.ini** with the new pacenotes inside **c:\temp\track\pacenotes\salida**

0.01 is a sensibility factor (0.01-0.03 is usually ok). The lower, the bigger the number of pacenotes generated. 10 is the length (number of anchors, usually spaced 5m) you want the pacenotes to be moved towards the start of the track (50m usually works fine)

A graph will pop-up showing the track and the decission made for each curve.

The pacenotes are generated based on 2 parameters: angle turned in the curve in the central 50m and the (global) angle turned in 100m. Criteria for assignation can be changed just editing the file c:\temp\track\pacenotes**asignar_pacenotes.m**. It is easy to change the file.

Explanation of how asignar_pacenotes.m works:
Only curves with a central angle turn bigger than 15 degrees and considered. If the 100m turn is bigger than 140 degrees it will be considered as a "hairpin", unless the central turn is smaller than 80 degrees, case in which the call will be "Medium Long". And so on.

X. More scripts. Pacenotes creation (to be inserted inside the Venue.xml)

This process is only useful for projects created with the scripts while we are still working with the Venue.xml (we still haven't made changes with BTB0.8 and saved), beceause generated pacenotes will be inserted inside that fila. This process requires the road having being splitted (Step 8, working inside folder s10_split).

> **cd f:\project\pacenotes**
> **coge_datos**
> **pacenotes_a**
> **pacenotes2_a(0.03,10)**

The output of this scripts is the file salida\pacenotes.txt. If it exists, join all will add it to the Venue.xml

XI. More scripts: walls on the boundaries of the road

(May be you should consider reading also “How add Sobjects with the scripts.doc”)

Script *muros_pegados* creates walls on both sides of the road, for its whole length. Walls can be moved a distance away from the road or towards its center.

> **cd c:\project\s7_walls_b**
> **muro_pegado(20,0.5)**

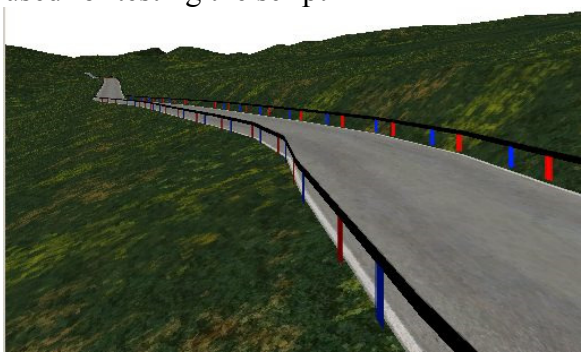
The first parameter is the number of road Anchors that will define each wall (default separation between Anchors is 5m). The second parameter is the displacement in meters of the walls in the outside direction. If it is negative the walls will move to the inside of the road.

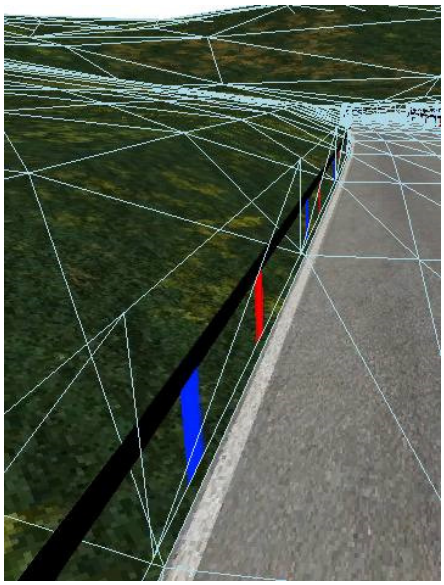
Script ouput are files **salida\muros_izda.txt** and **salida\muros_dcha.txt**, that contain the walls for the left and right sides, respectively.

NOTE: if the project already has more walls (invisible protection walls, for example) the total amount of walls must be updated at the beginning of the *Walls* section of the Venue.xml.

NOTE: characteristics (profile, materials, etc.) of the walls are defined by files final_muro2.txt and final_muro2_invertido.txt. Chanaging those files the characteristics of all the created walls will be hereafter different.

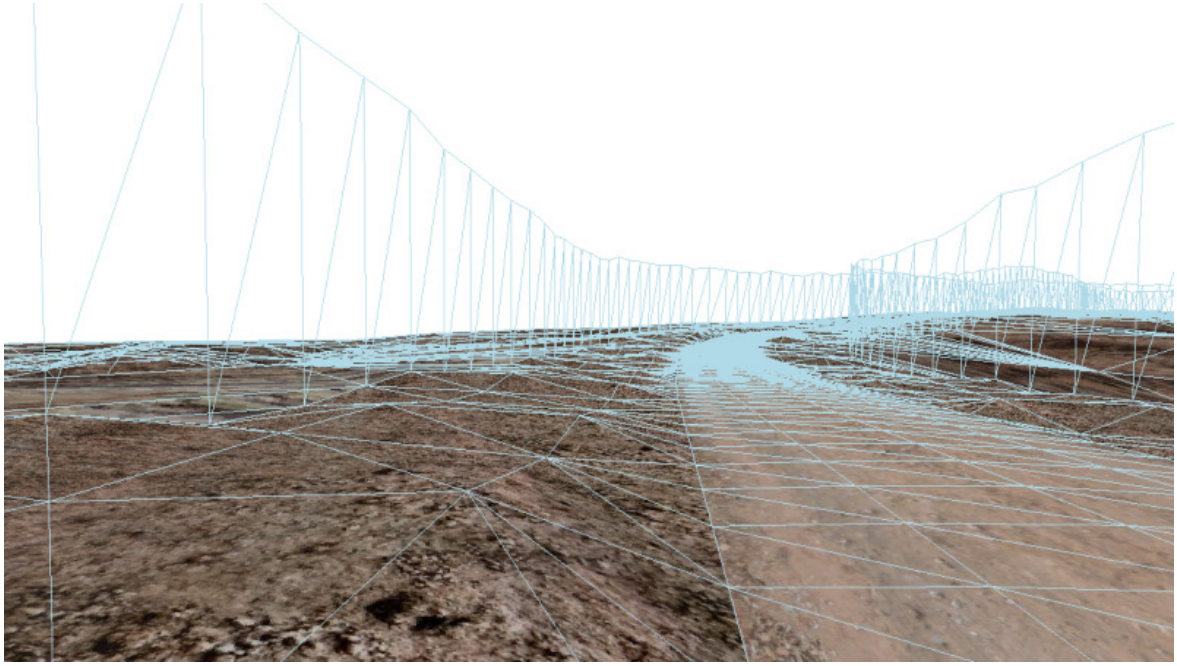
NOTE: there is a link in the forum for downloading XPack TestMuros, the Xpack I used for testing the script





XII. More scripts: making the terrain bumpier

You have finished using the scripts for creating a basic BTB project. You are satisfied, but the terrain is too flat and you wish you had more bumps on it.
You want the terrain to be more irregular, more uneven.



So you can use the script **terrain_noise**

```
>cd c:\project\s4_terrain
>terrain_noise([-0.4,0.4])
```

Where you say you want terrain anchors (nodes) elevation to change a random value between -0.4 and 0.4 meters

Then substitute, inside the Venue.xml, TerrainAnchors section with the new c:\project\s4_terrain\listado_anchors.txt