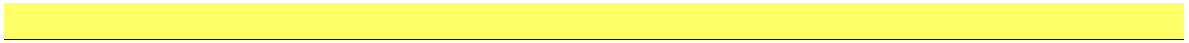

Guía del método Zaxxon



Index

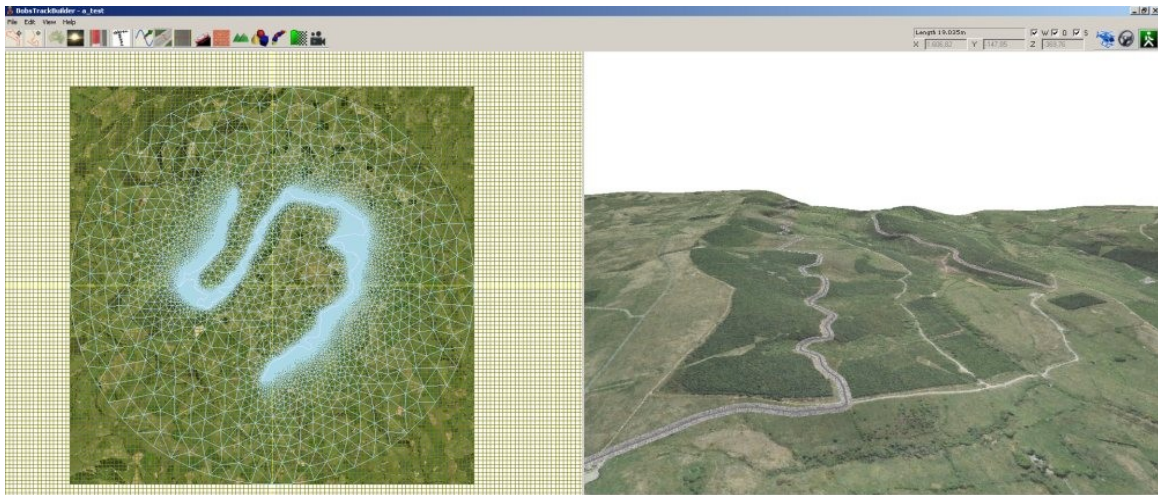
| | | |
|--------|---|----|
| I | Resumen | 1 |
| I.1 | ¿Qué es el "método zaxxon"? | 1 |
| II | Instalación del software | 3 |
| II.1 | La estructura de carpetas y los scripts | 3 |
| II.2 | Octave 3.2.3 | 5 |
| II.3 | El interfaz gráfico (GUI) para los scripts | 5 |
| II.4 | Instalación de gmsh | 7 |
| II.5 | WP.zip | 7 |
| III | Antes de usar el método | 8 |
| III.1 | One-track o multi-track? | 8 |
| III.2 | ¿Origen de los datos de elevación? | 9 |
| IV | Usando los scripts | 13 |
| IV.1 | Proyecto multi-track | 13 |
| IV.2 | Proyectos One-track | 23 |
| V | Trabajar con diferentes fuentes de datos de elevación | 24 |
| V.1 | Google Earth | 24 |
| V.2 | AGR Data | 25 |
| V.3 | Seamless server | 28 |
| V.4 | Ficheros hgt | 29 |
| VI | Ajuste fino del perfil en altura de la carretera | 30 |
| VI.1 | Retocar a mano el perfil | 30 |
| VI.2 | Uso del script "corregir" | 32 |
| VII | Usando gmsh | 36 |
| VII.1 | ¿Qué son anchors_carretera.geo y joined.geo? | 36 |
| VII.2 | Conceptos basicos de gmsh | 36 |
| VII.3 | Controles básicos de gmsh | 36 |
| VII.4 | ¿Cuáles son los límites de la zona no conducible? | 37 |
| VII.5 | ¿Cómo debo crear la frontera externa del terreno no conducible? | 37 |
| VII.6 | ¿Cuál es la salida del proceso? | 38 |
| VII.7 | ¿Cómo se definen las Physical Surface? | 39 |
| VII.8 | Gmsh thresholds | 41 |
| VII.9 | Processing joined.geo with gmsh | 45 |
| VIII | Background images | 58 |
| VIII.1 | Basics | 58 |
| VIII.2 | Blending with background images | 59 |
| VIII.3 | Resumen | 61 |
| IX | Using LiDAR data | 65 |
| X | Advanced uses of the scripts | 70 |
| X.1 | Terrain that matches background images | 70 |
| XI | List of commands | 72 |
| XII | Additional Notes | 80 |

| | | |
|--------|--|----|
| XII.1 | GUI | 80 |
| XII.2 | Gmsh and multitrack | 80 |
| XII.3 | Memory exhausted | 83 |
| XII.4 | Gmsh crashes while meshing 2D? | 83 |
| XII.5 | Can I add a kml route to gmsh? | 83 |
| XII.6 | Translate tool in gmsh | 84 |
| XIII | Links | 86 |
| XIII.1 | Videoutorials | 86 |
| XIII.2 | Forums | 87 |
| XIII.3 | Old help files | 87 |
| XIII.4 | parlesportes' site | 87 |
| XIV | License | 88 |

I RESUMEN

I.1 ¿Qué es el "método zaxxon"?

Básicamente es una forma "rápida" de crear proyectos base para BTB, utilizando para ello datos de elevación. Una vez te acostumbras al método puedes tener un proyecto BTB preparado para ser editado con el BTB en unas horas. Se puede incluir terreno y carreteras creados a partir de datos de elevación, y si se quiere texturizados con imágenes de satélite reales.



Un resumen simplificado del método, teniendo en cuenta que hay múltiples formas de hacer las cosas sería:

1. Todo empieza con el kml de la ruta. Se convierte a coordenadas BTB
2. Se obtienen datos de elevación suficientes para cubrir todas las carreteras y terreno.
3. Se le da a la carretera un perfil en altura acorde con los datos de elevación que se tienen (se puede retocar a mano)
4. Entonces se definen los límites del terreno alrededor de las carreteras y se crea un mallado para ese terreno.
5. Se le da elevación al terreno a partir de los datos de altura de que se han conseguido
6. Se crean muros invisibles para evitar que el coche salga del terreno conducible (solo para RBR)
7. Se dividen las carreteras en segmentos por cuestión de eficiencia tanto en BTB

como in-game

8. Se divide el terreno empleando una rejilla $m \times n$ (también por eficiencia)
9. Se crea un Venue.xml, un fichero que BTB puede leer y que permite empezar a trabajar con el proyecto

Se pueden incluir automáticamente imágenes de fondo (satélite) en el proyecto. Quizá únicamente para tenerlas de referencia o para usarlas como textura textura para el terreno.

II INSTALACIÓN DEL SOFTWARE

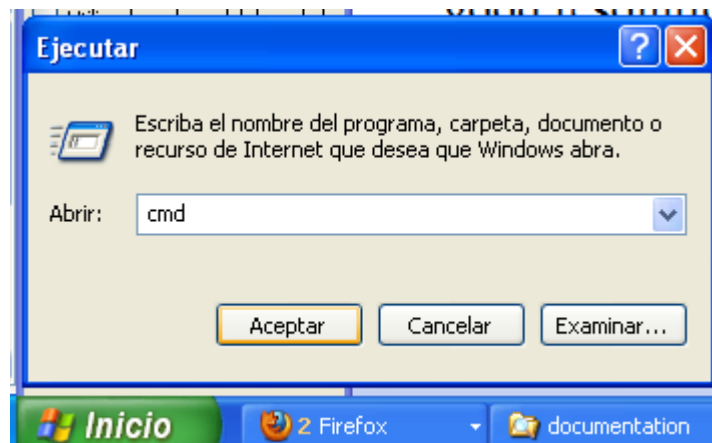
Para usar los scripts se necesita:

1. [La estructura de carpetas y los scripts](#)
2. [Octave 3.2.3](#)
3. [El interfaz gráfico \(GUI\) para los scripts](#)
4. [Instalación de gmsb](#)
5. [WP.zip](#)

II.1 La estructura de carpetas y los scripts

Se necesita Subversion (SVN) para descargar los ficheros. Se pueden descargar clientes gratuitos visitando: <http://subversion.apache.org/packages.html> (por ejemplo <http://www.sliksvn.com/pub/Slik-Subversion-1.7.2-win32.msi> from <http://www.sliksvn.com/pub>)

Una vez has instalado el cliente Subversion, en Windows abre una consola de texto: Inicio->Ejecutar-> "cmd"



En la consola escribe:

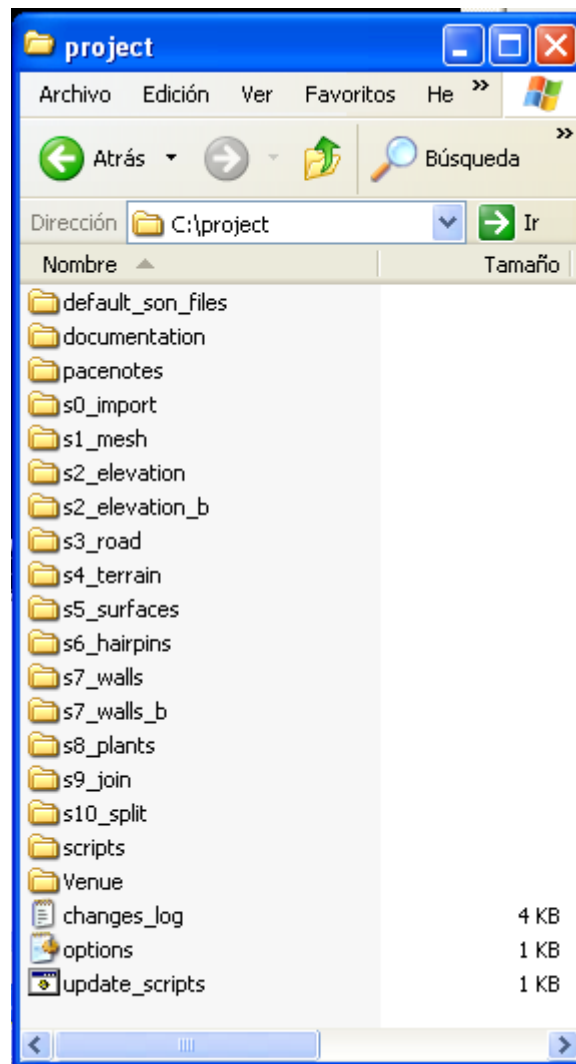
```
svn checkout http://subversion.assembla.com/svn/zaxxon_scripts/trunk c:\
```

(O se puede descargar [download_scripts.bat](#) y ejecutarlo (doble click))

```
C:\WINDOWS\system32\cmd.exe - svn checkout http://subversion.assembla.com/svn/zaxxo...
Microsoft Windows XP [Versión 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\User>svn checkout http://subversion.assembla.com/svn/zaxxon_scripts/trunk c:\
```

Se obtendrá una copia de los ficheros necesarios en **c:\project**

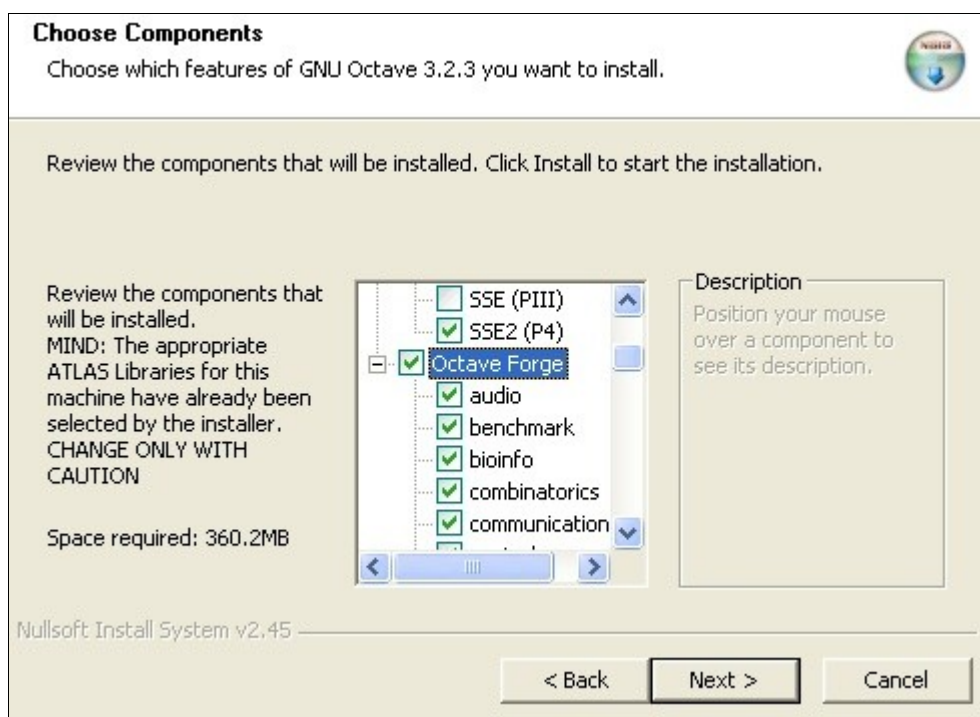


II.2 Octave 3.2.3

Se necesita [Octave 3.2.3](http://sourceforge.net/projects/octave/files/Octave_Windows%20-%20MinGW/Octave%203.2.3%20for%20Windows%20MinGW32%20Installer/Octave-3.2.3-2_i686-pc-mingw32_gcc-4.4.0_setup.exe/download) (NO instalar una versión diferente de Octave):

http://sourceforge.net/projects/octave/files/Octave_Windows%20-%20MinGW/Octave%203.2.3%20for%20Windows%20MinGW32%20Installer/Octave-3.2.3-2_i686-pc-mingw32_gcc-4.4.0_setup.exe/download

OBLIGATORIO: durante la instalación **marca Octave Forge libraries**



Tras la instalación de Octave 3.2.3, abrirlo y ejecutar:

> pkg rebuild -auto communications

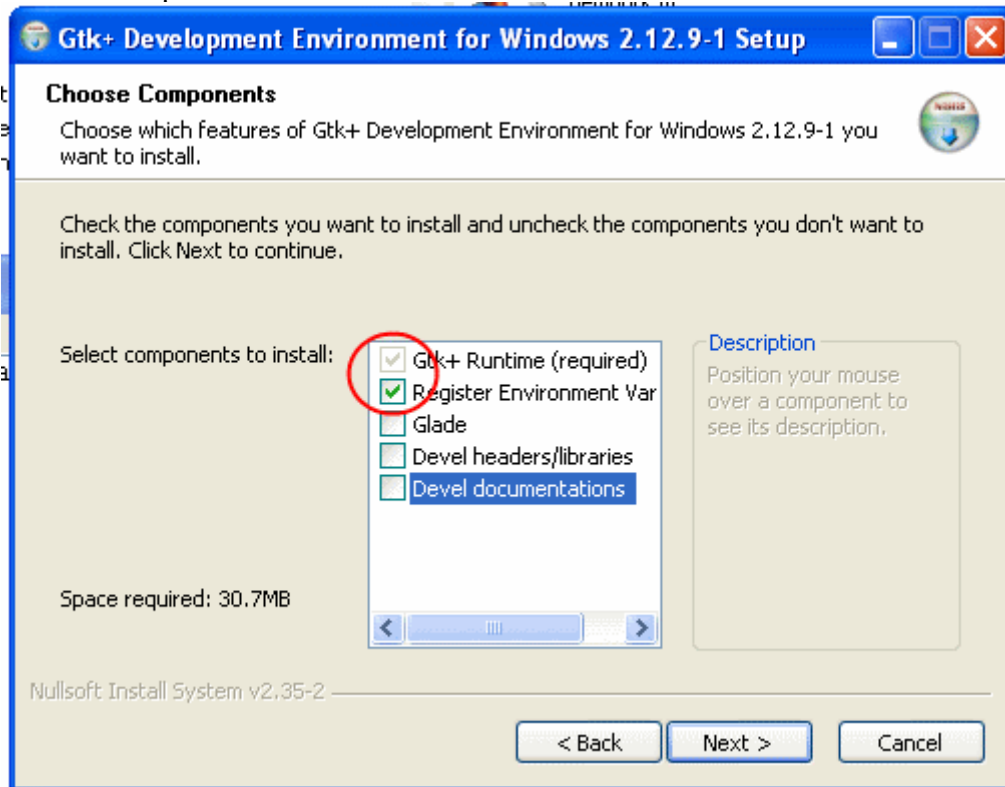
Luego cerrar Octave y abrirlo de nuevo cuando se quiera comenzar a usarlo.

II.3 El interfaz gráfico (GUI) para los scripts

1. Instalar GTK (gtk-dev-2.12.9-win32-2.exe)

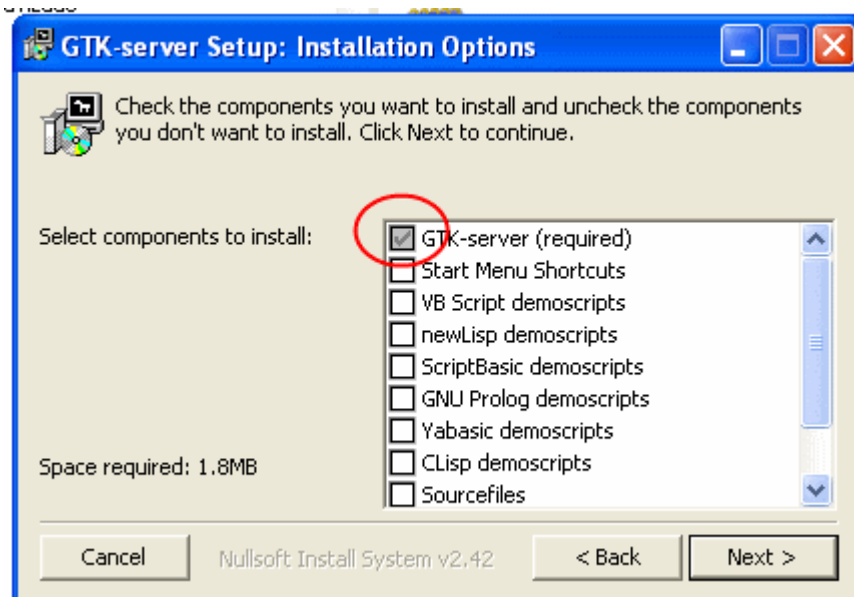
<http://sourceforge.net/projects/gladewin32/files/gtk%2B-win32-devel/2.12.9/gtk-dev-2.12.9-win32-2.exe/download>

Marcamos las dos primeras casilla:

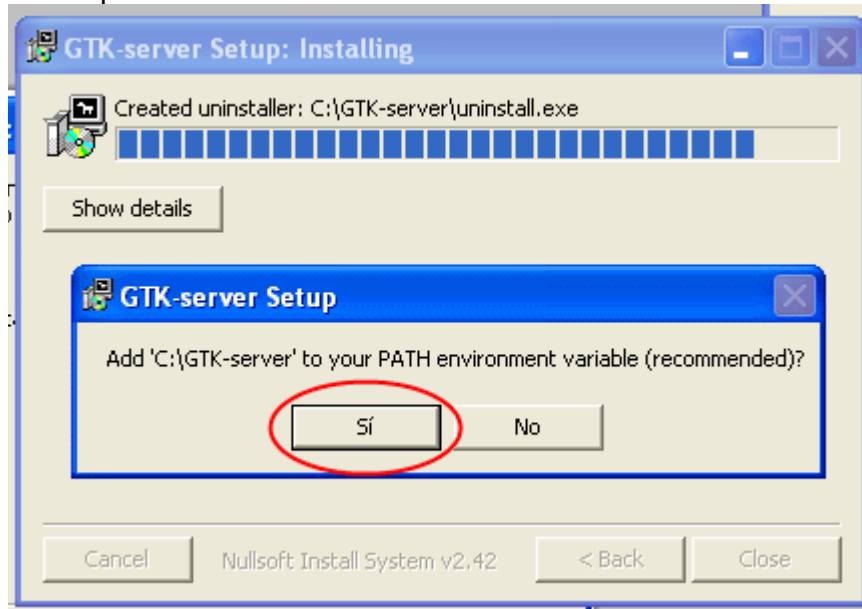


2. Instalar gtk-server (gtk-server-2.3.1-installer.exe)

<http://downloads.sourceforge.net/gtk-server/gtk-server-2.3.1-installer.exe>



Seleccionamos la opción “add to the PATH”:



II.4 Instalación de gmsh

Gmsh es un generador de mallados para elementos finitos. Lo usaremos para crear un mallado para nuestro terreno. Debes instalarlo. Basta descargarlo y extraer its el contenido en la carpeta que elijas.

Gmsh: <http://geuz.org/gmsh/#Download>

II.5 WP.zip

El proyecto creado usará tecturas de un XPack de nombre WP.zip. Se puede conseguir el XPack WP.zip de: <http://www.mediafire.com/?z2tcoh4wyxj>

III ANTES DE USAR EL MÉTODO

III.1 One-track o multi-track?

¿Cuántas rutas .kml tienes para las carreteras de tu proyectot? Si solo tienes una ruta hablaremos de un “proyecto one-trackt” y usar los scripts será muy sencillo. Si tienes más de una ruta (desvíos, cruces, rutas alternativas, etc.) el proceso es un poco más complicado, especialmente a la hora de usar gmsh.

One-track

En los proyectos one-track no hay que hacer nada especial con los ficheros que tenemos en [c:\project](#). Estás listo para empezar a usar los scripts.

Multi-track

Cuando se trabaja con varias rutas los scripts se usan básicamente de la misma forma en todas ellas, cada una procesada dentro de su propia estructura de carpetas. El trabajo que se hace por separado luego se combina, creando un proyecto que incluye rutas diferentes/independientes, como desvíos sin salida (uso estético) o rutas que pueden usarse como caminos alternativos.

Una idea importante es que debemos elegir una ruta principal. A esa ruta la llamaremos el FATHER. Y las rutas secundarias (todas las demás) serán los SONS.

Para preparar un proyecto multi-track:

1. Mover el contenido de c:\project to c:\project\father
2. Si por ejemplo se tienen tres sons, abrir octave y ejecutar:

```
○ > addpath('c:\project\father\scripts')  
○ > cd c:\project\father  
○ > create_sons(3)
```

Ahora tendremos los ficheros del father en **c:\project\father**, y los ficheros de los 3 sons en **C:\project\son01**, **C:\project\son02** and **C:\project\son03**. Eso es todo: ya estamos listos para emplear los scripts.

NOTE: los sons saben que tienen un father y su ubicación gracias a que en sus carpetas se ha creado un fichero **father.txt** con el contenido “father”. El father sabe que tiene 3 sons y su ubicación porque en su carpeta se ha creado un fichero llamado “**sons.txt**”, de contenido:

son01

son02

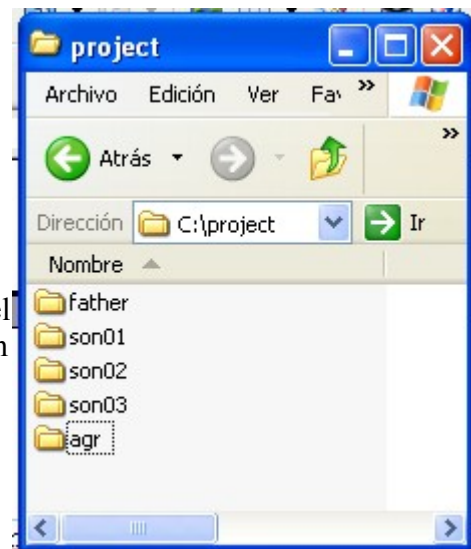
son03

III.2 ¿Origen de los datos de elevación?

Las fuentes habituales de datos de elevación son Google Earth y AGR (ASCII Grid). El GUI está preparado para usar solo estas dos fuentes. Si se quiere usar otra, se debe leer el capítulo V.

AGR

- 1) Crear una carpeta llamada “agr” en la misma carpeta en la que está “project” (o las carpetas del father’s y los sons en los proyectos multi-track). Ejemplo: si trabajamos en un proyecto one-track en c:\project, crearemos la carpeta c:\agr.
- 2) Copia los ficheros .agr a la carpeta creada en el paso 1). Renombra esos ficheros con extensión .agr si es necesario.
- 3) Usa los scripts con normalidad.



NOTA: si los ficheros .AGR son demasiado grandes los scripts pueden no ser capaces de usarlos. Se puede emplear el script **split_agr** para dividir un fichero en ficheros más pequeños, como se explica en la sección V.2

Google Earth

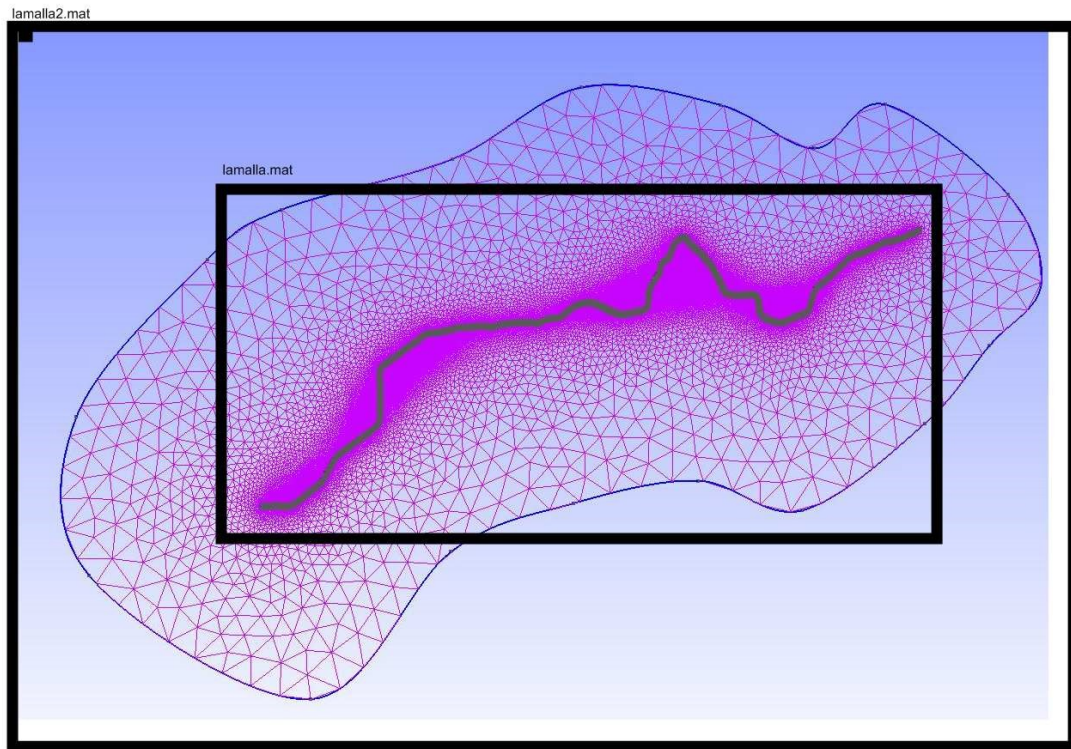
Se pueden conseguir datos de elevación de Google Earth (leer sección V.1 para ampliar). En ese caso los scripts crearán una rejilla de puntos, distribuirán esos puntos en decenas de ficheros (de nombre `gridXXX.kml`) y pedirán a GE que les de elevación. Los ficheros se procesarán uno tras otro, a mano si se usa un programa del estilo de BTBLofty o automáticamente si se usa python).

Los scripts pueden trabajar con una única rejilla de datos. Pero se recomienda usar 2. La razón para recomendar usar 2 rejillas es que queremos buena resolución para los datos cerca de las carreteras **pero es absurdo tener una resolución de 10m a 1Km de las carreteras** porque el mallado que creemos tendrá triángulos muy grandes (75-250m de lado) lejos de las carreteras. Por tanto podemos crear dos rejillas y buscar datos de elevación para ellas: una rejilla con buena resolución que debe cubrir por completo las carreteras (father y sons) y otra con peor resolución cubriendo por completo el terreno del proyecto.

La rejilla pequeña se creará dentro de la carpeta `s2_elevation`. Para la rejilla grande se empleará otra carpeta: `s2_elevation_b`. Los datos de altura se copiarán automáticamente a la carpeta `s4_terrain` con el nombre **lamalla2.mat**. Para la rejilla pequeña el nombre usado será **lamalla.mat**.

Como ya se ha dicho, los scripts también trabajarán si existe una única rejilla (creada en `s2_elevation`), pero como dicha rejilla debería cubrir todo el terreno, si tuviese que tener una buena resolución espacial, el número de puntos de la rejilla sería enorme, al igual que el fichero `lamalla.mat` (y consecuentemente el tiempo requerido para obtener los datos de altura de Google Earth).

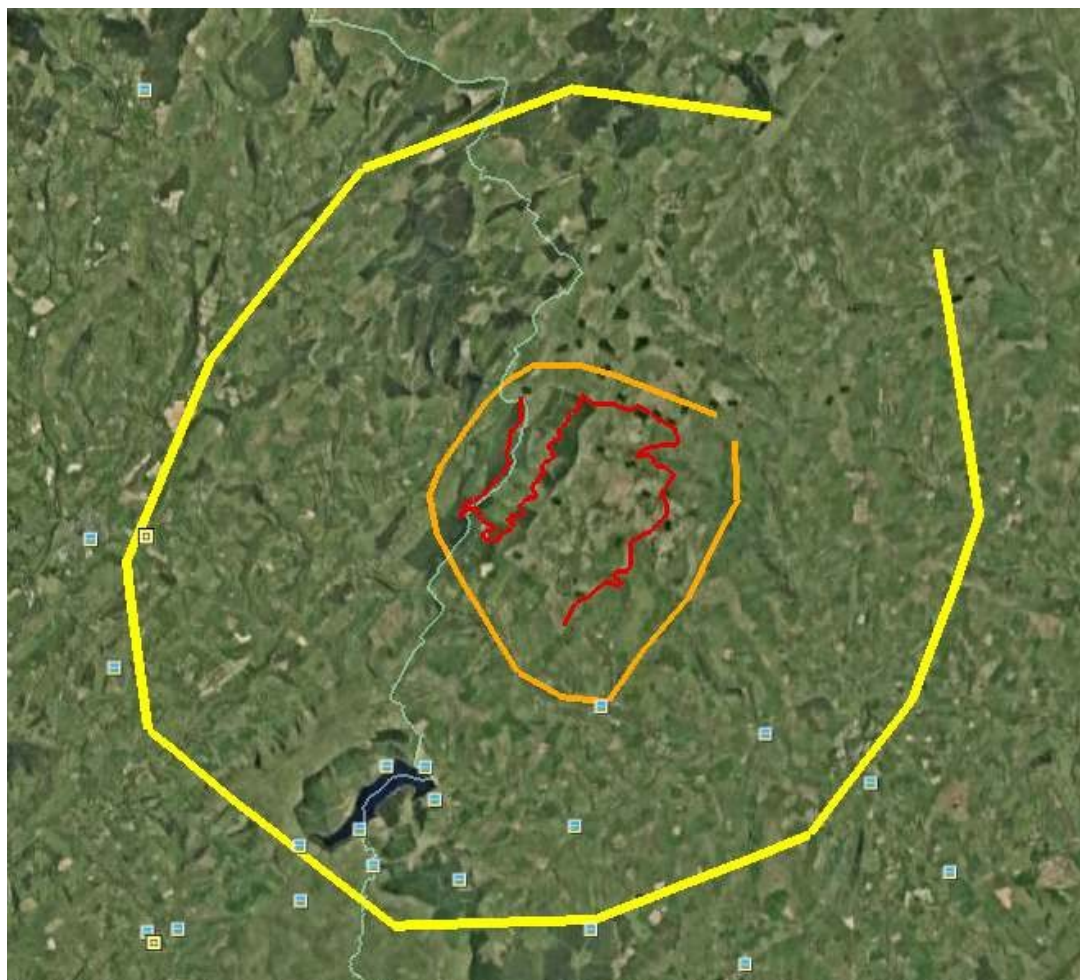
La siguiente imagen es ilustrativa:



La mejor forma de delimitar ambas rejillas es crear rutas con Google Earth. Creamos dos kmls más:

- 1) "**limits.kml**" (ruta naranja) rodeando la carretera principal y todos los desvíos. Se debe grabar en **s2_elevation**
- 2) "**limits_b.kml**" (ruta amarilla) delimitando el terreno deseado para el proyecto. Se debe grabar en **s2_elevation_b**

Las rutas no necesitan ser cerradas y los scripts crearán un par de rejillas suficientemente amplias como para cubrirlas por completo. La rejilla grande cubrirá todo el terreno encerrado por la ruta amarilla y la rejilla pequeña cubrirá todo el terreno rodeado por la ruta naranja. El color de la ruta es indiferente para los scripts y solo se menciona para ayudar a identificarlas.



IV USANDO LOS SCRIPTS

IV.1 Proyecto multi-track

NOTA: Llegados a este punto se debe haber copiado los .kml del father y los sons dentro de sus respectivas carpetas s0_import. La elevación de los puntos del kml será ignorada.

Y finalmente abrimos octave, añadimos la carpeta donde están los scripts a los paths donde octave buscará los comandos y ejecutamos **zgui**:

```
> addpath('c:\project\father\scripts')
> cd c:\project\father
> zgui
```

zgui es el interfaz para la primera parte de uso de los scripts (antes de usar gmsb)

zgui(1) es el interfaz para la segunda parte de uso de los scripts (cuando ya existe anchors_carretera.msh)

NORMAL y FATHER tienen 2 interfaces: zgui y zgui(1). Los SONS solo tienen zgui, pero no zgui(1).

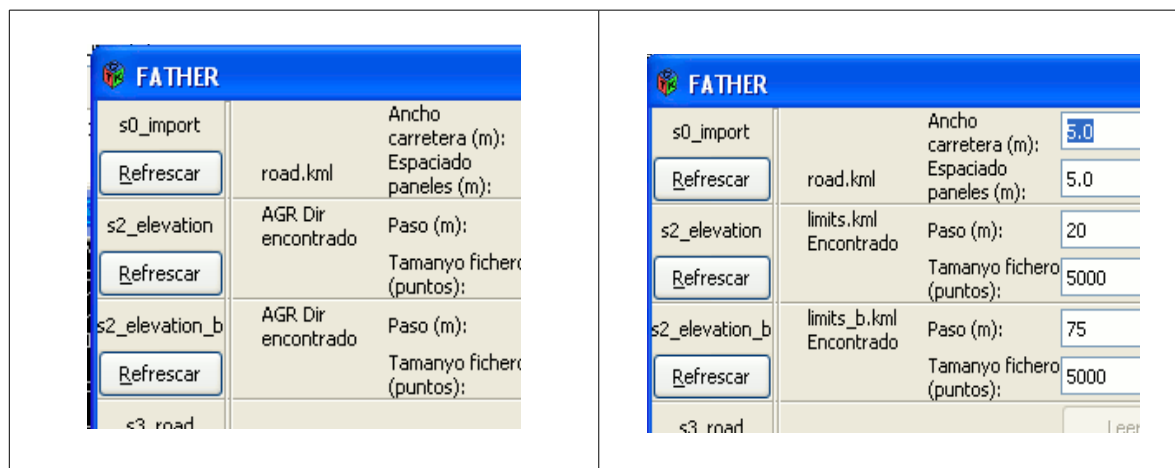
Se debería abrir una nueva ventana en nuestro sistema, el GUI:

Para comenzar:
Copia el kml en el directorio s0_import
y pulsa "Importar el kml"

For more info Visit:
<http://www.plpcreation.tonsite.biz/>
<http://forum.rallysim.fr/viewtopic.php?f=51&t=3025&start=825>
<http://btbtracks-rbr.foroactivo.com/t131-zaxxon-s-method-tutorial>
<http://foro.simracing.es/bobs-track-builder/3815-1.html>
<http://devtrackteam.solorally.it/viewforum.php?f=28>

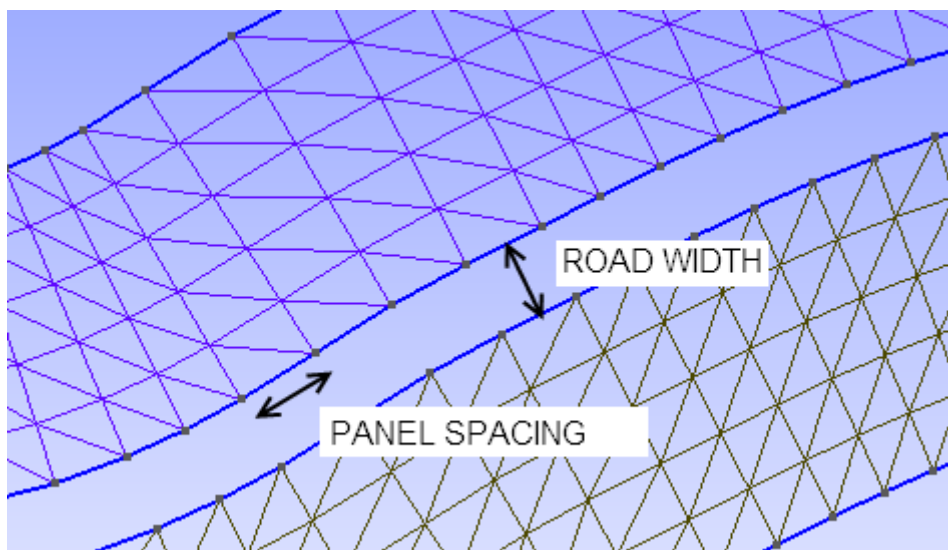
En la barra superior debería leerse la palabra “FATHER”. Y si estamos usando datos AGR

deberíamos ver un mensaje en la ventana diciendo “AGR Dir encontrado”. En ese caso los pasos s2_elevation y s2_elevation_b deben saltarse. En caso contrario, si se usan datos de altura de Google Earth un mensaje en pantalla debe informarnos de que los ficheros “limits.kml” y “limits_b.kml” han sido encontrados.



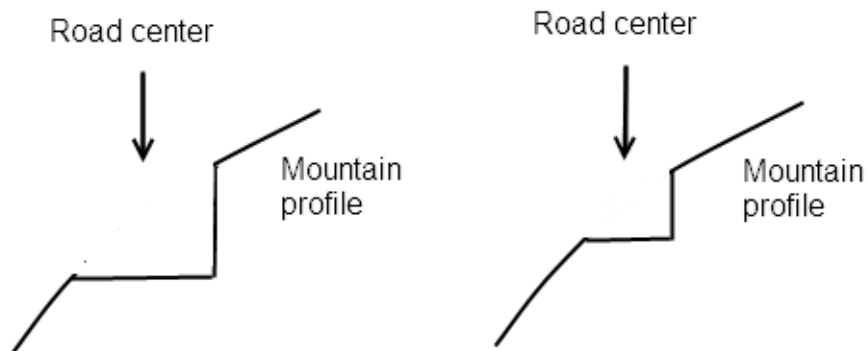
s0_import

El próximo paso es pulsar el botón “Importar el kml”. Para este botón necesitamos especificar un ancho para la carretera y el espaciado que queremos entre los puntos que unen la carretera y el terreno.



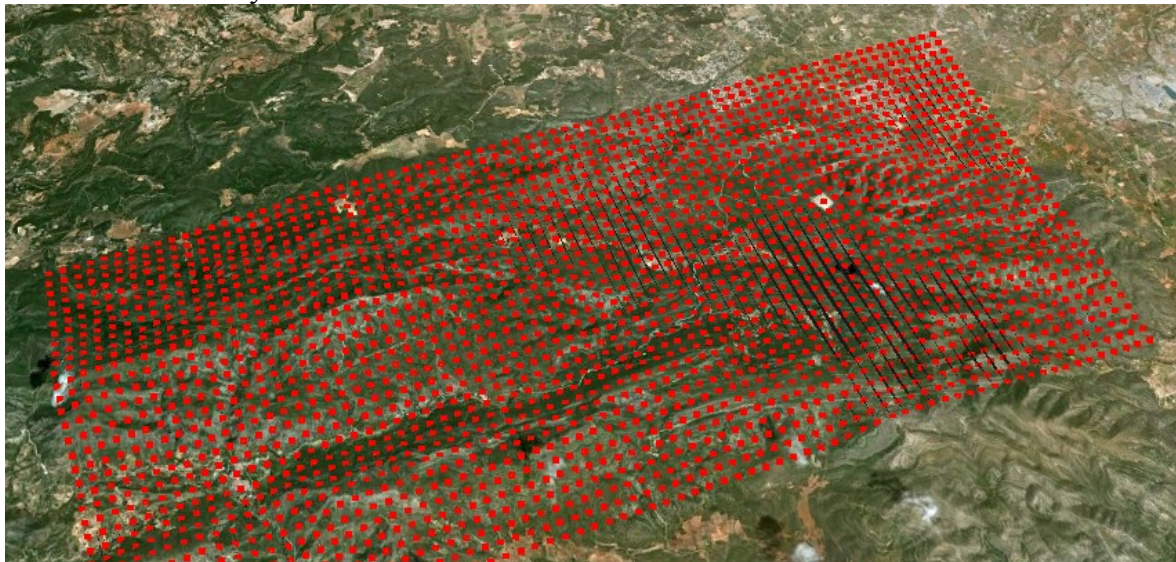
En este paso cada punto del .kml se traducirá en un nodo de la carretera BTB. Si el paso falla, se debería revisar el kml buscando pequeños lazos (puntos que retroceden, en lugar de avanzar) o puntos demasiado próximos.

El ancho de la carretera es importante porque afectará al mallado que vamos a crear pero también porque el perfil en altura de la carretera depende del ancho de la misma. Cuando los datos de altura no son buenos a una carretera ancha los scripts le asignarán una altura menos que a una carretera estrecha. Posiblemente el mejor valor para el ancho de la carretera sea uno próximo al ancho real de la misma.



s2_elevation/s2_elevation_b

El siguiente paso es conseguir datos de altura, si se está usando GE como fuente de dichos datos. En ese caso hay que definir la separación entre los puntos de la rejilla y pulsar “Crear la rejilla”. Los puntos de la rejilla se distribuirán en varios ficheros (llamados gridXXX.kml) que se crearán dentro de s2_elevation\salida o s2_elevation_b\salida. El tamaño del fichero es la cantidad de puntos incluidos en cada fichero. 5000 es el máximo recomendado. Si se usa un valor superior a 5000 Google Earth se ralentiza drásticamente. Valores entre 1000 y 5000 son una buena elección.

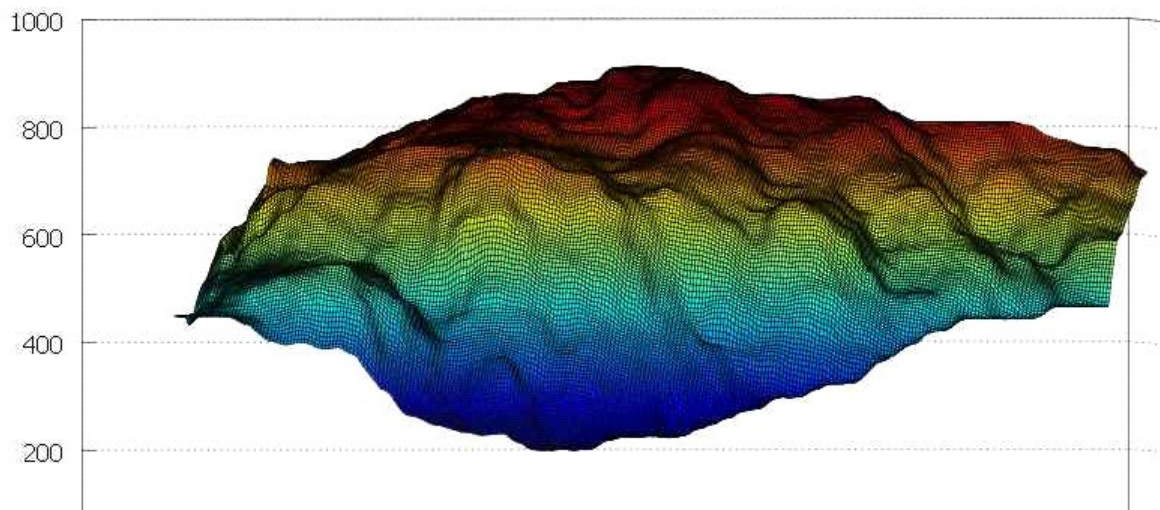


Ahora, si has instalado python (leer la sección V.1) puedes pulsar en el botón “Elevar los kmls”. En caso contrario tendrás que elevar los ficheros empleando otra aplicación, como

las listadas en la sección V.1.

Cuando el proceso finaliza cada fichero gridXXX.kml ha sido procesado y se ha creado un fichero gridXXX_relleno.kml con los mismos puntos pero con elevación. El siguiente paso es leer los ficheros gridXXX_relleno.kml y recopilarlos en la matriz de datos que usarán el resto de scripts. Haz click en el botón “Leer rejilla”. Un fichero llamado lamalla.mat se creará en la carpeta s2_elevation\salida.

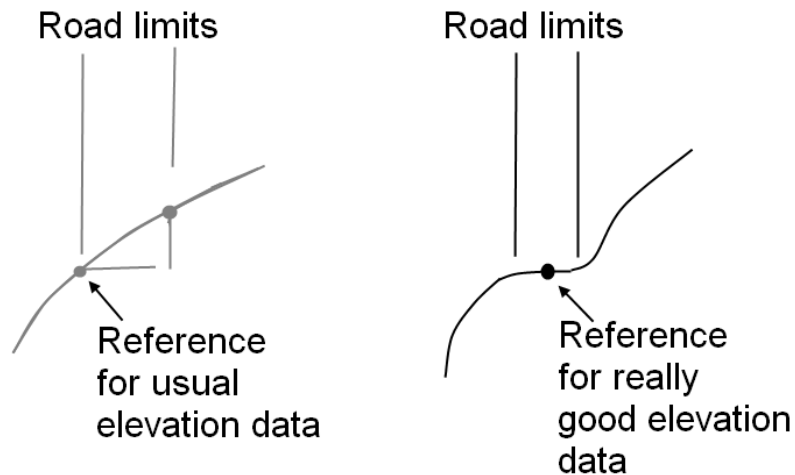
Puede ser buena idea en este momento pulsar el botón “Ver datos” y comprobar que a simple vista los datos recopilados no contienen errores.



s3_road

Ahora queremos darle un perfil en altura a la carretera. Para las fuentes de datos habituales el primer paso es calcular la altura del terreno a ambos lados de donde se sitúa la carretera (en concreto en los puntos que llamamos “anchors”). Cuando el terreno tiene distinta elevación a ambos lados de la carretera los scripts usan el más bajo de los dos como referencia para crear el perfil de elevación de la misma. No obstante si tuviésemos datos de elevación de elevada resolución (un dato cada 1m) podríamos usar el centro de la carretera como referencia de altura. Este sería el caso de emplear datos LiDAR.

Ante la duda NO marque “Usar la línea central”.



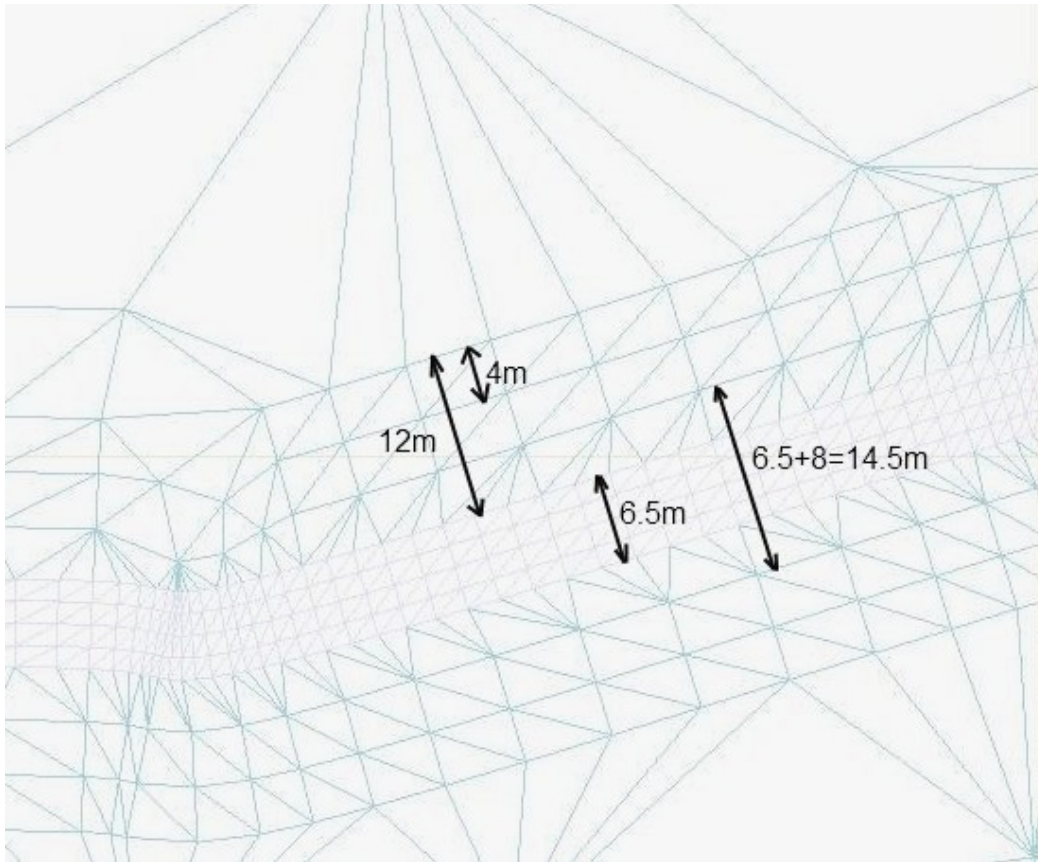
Una vez tenemos referencias de altura a lo largo de toda la carretera, las usamos para crear un perfil en altura suave para la misma. Se empleará un factor de suavizado S (S debe ser siempre impar) para crear un nuevo juego de valores de altura donde cada valor es el promedio de los S valores más próximos a ese punto. La distancia tenida en cuenta en el suavizado depende de la separación entre anchors. Por defecto la separación entre anchors es de 5m, pero se ha podido escoger otro valor en el paso `s0_import`. La mejor opción es probar valores (pulsando “Crear perfil en altura”) y examinar el resultado. El otro parámetro limita la pendiente de la carretera que se va a crear. 0.25 significa 25%. Se puede poner a 1 si no se quiere emplear este parámetro (1 sería limitar a una pendiente del 100%, y eso en una carretera normal no debería tener efecto).

Si te satisface el perfil en altura creado debes pulsar "e" en la ventana de texto de octave y luego <ENTER>. Finalmente hay que consolidar el perfil pulsando el botón “Consolidar”, el último del paso `s3_road`. Si no nos gusta el perfil, en este paso se permite al usuario ajustar a mano el perfil en altura, e incluso cambiar un poco los datos de altura (solo si se usan ficheros `lamalla.mat`, no en otros casos como AGR) tratando de adaptarlos a la carretera que queremos. Leer sección VI para detalles.

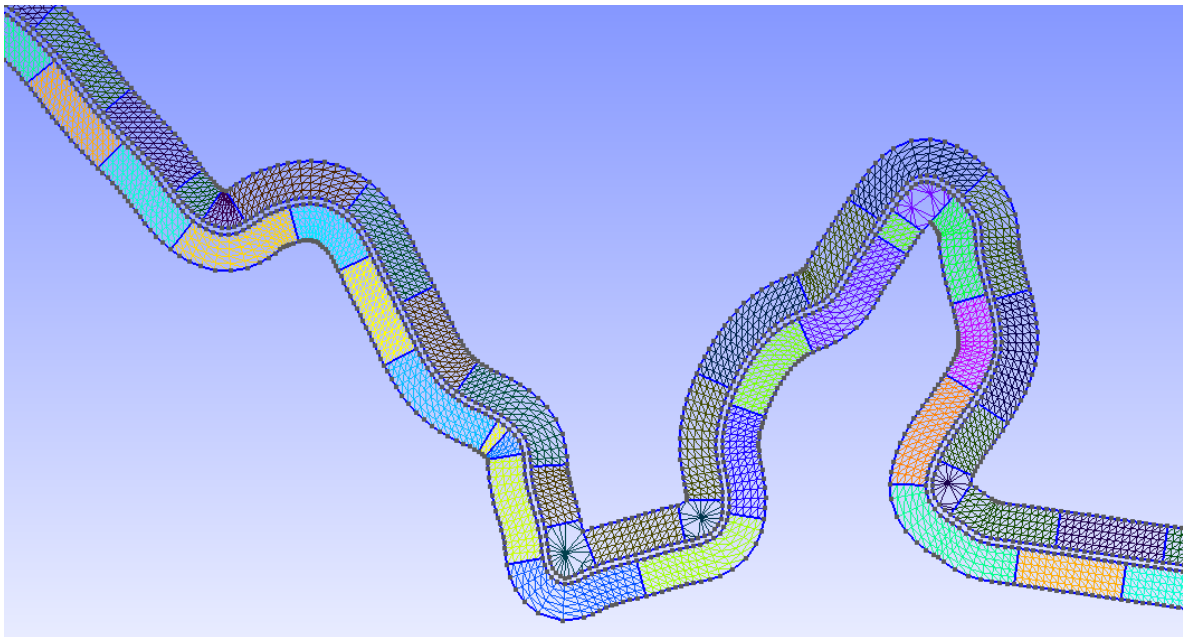
s1_mesh

Una vez tenemos nuestra carretera, el siguiente paso es crear terreno conducible a ambos lados de la misma. Tenemos que indicar dos datos: el ancho total de terreno conducible y cuántos paneles hay en dicho ancho. Hay una opción para dividir automáticamente la superficie conducible creado por los scripts en segmentos más cortos.

La siguiente imagen puede ser de ayuda para entender los parámetros: se usa un ancho de 12m y 3 panels. La malla creada por los scripts para la parte conducible será la base para que el usuario cree otra malla para la parte no conducible. El usuario tiene que crear esa malla usando `gmsh`.

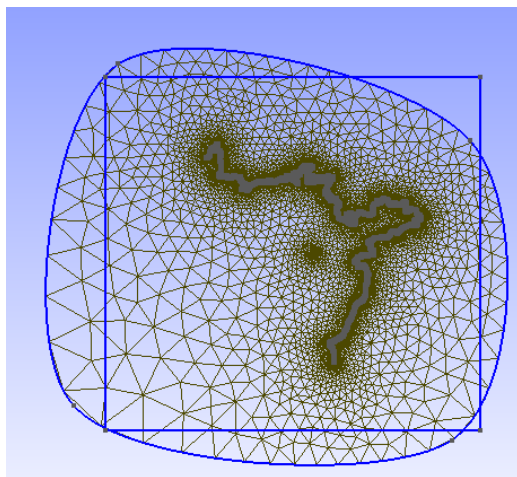


En el caso de que se divida la superficie conducible en segmentos más cortos se obtiene algo como esto:

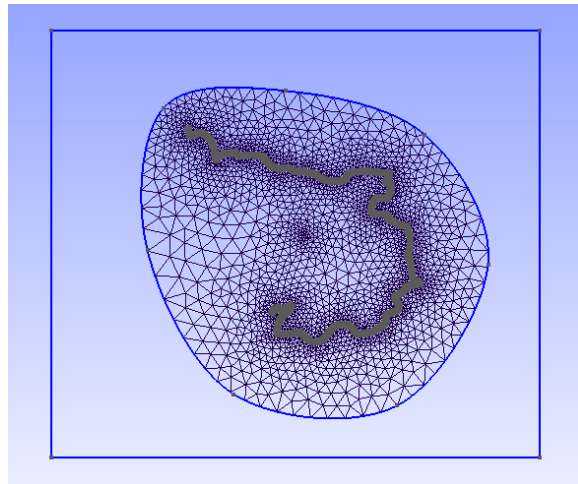


El parámetro MINISUPS del fichero options.ini ajusta cada cuántos puntos queremos una nueva superficie.

En este paso es realmente importante seleccionar “**Incluir límites de elevación**” en caso de que se use Google Earth como fuente de datos de altura. Eso debería ayudarnos a no crear un mallado que exceda los datos de elevación disponibles, un mallado que no podría ser procesado por los scripts. Por ejemplo, en la parte izquierda de la siguiente figura se muestra un mallado que excede los límites y que por tanto debe ser rehecho para que sus límites no se salgan del rectángulo (límites de datos de elevación disponibles).



NO OK



OK

No sigas con el proceso a menos que estés seguro de que el mallado está cubierto por completo por los datos de elevación.

Una vez que hemos procesado al father (al menos parcialmente), tenemos que procesar los sons. Los pasos son similares pero no idénticos, pues en los sons no se gestionan datos de altura.

```
> cd ..  
> cd son01  
> zgui
```

La ventana del GUI es similar a la del FATHER, pero deberíamos leer “SON” en la barra superior.

The screenshot shows the SON software interface with the following sections:

- s0_import**: Includes a 'Refrescar' button, a file path 'son01.kml', and input fields for 'Ancho carretera (m):' (set to 5.0) and 'Espaciado paneles (m):' (set to 5.0). There is an 'Importar el kml' button.
- s3_road**: Includes a 'Refrescar' button, a 'Leer datos elevacion' button, a 'Factor de suavizado:' input (set to 15), a 'Pendiente maxima:' input (set to 0.25), and buttons for 'Crear perfil en altura', 'Consolidar', and 'Corregir terreno (opt)'. There is also an unchecked checkbox for 'Usar linea central'.
- s1_mesh**: Includes a 'Refrescar' button, input fields for 'Ancho (m):' (set to 20) and 'nn Paneles:' (set to 5), a 'Crear el .geo' button, and checkboxes for 'Forzar regularidad' (unchecked) and 'Incluir limites de elevacion' (checked).
- s10_split**: Includes a 'Refrescar' button, input fields for 'nn segmentos:' (set to 1) and '0 m', a 'Crear puntos de ruptura' button, a 'Trocear carretera' button, a 'Dividir terreno con rejilla:' dropdown menu, an unchecked checkbox for 'Mezcla con fondo', and a 'Crear el terreno' button.

At the bottom, there is a text area with instructions: 'Para comenzar: Copia el kml en el directorio s0_import y pulsa "Importar el kml"' and a list of links for more information.

El SON tiene el paso “s10_split”, un paso sencillo en el que se trocea la carretera en varios trozos por motivos de eficiencia. Segmentos de alrededor de 1Km parecen funcionar bien. Mayores longitudes pueden hacer que tanto BTB como RBT vayan realmente lentos.

A veces los SONs son realmente cortos y usar factores de suavizado normales en el paso s3_road hace que los scripts fallen (si por ejemplo el SON tiene solo 7 anchors y seleccionamos un factor de suavizado de 13 no habrá suficientes puntos para hacer los cálculos). Seleccionar un factor de suavizado pequeño (incluso 1 llegado el caso) a menudo resuelve el problema.

Tenemos que **procesar todos los SONs**.

Una vez hayamos procesado todos los SONs tenemos que juntar todas las rutas: el FATHER y los SONs:

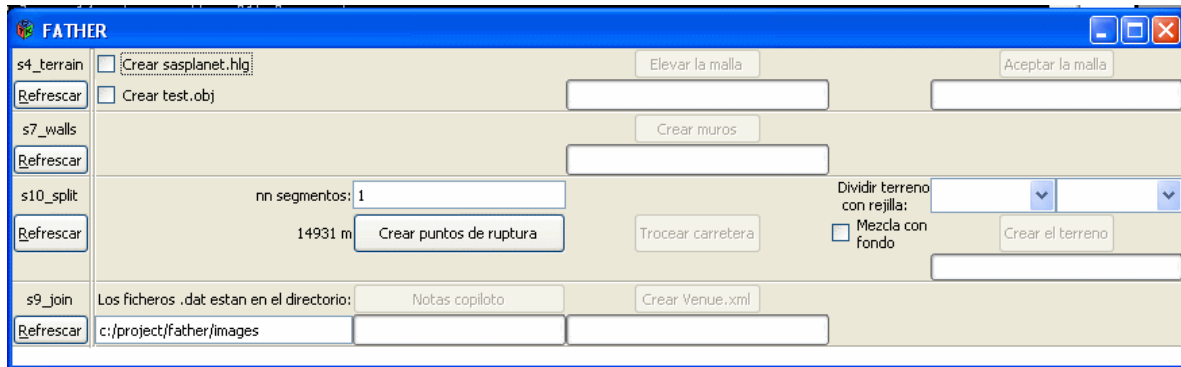
```
> cd c:\project\father\s1_mesh
> join_geos
```

La salida es c:\project\father\s1_mesh\joined.geo, un fichero que debe ser procesado con gmsh para crear el fichero .msh que necesitamos.

Procesar joined.geo no es un paso trivial así que no se explica en esta sección. La salida del proceso debería ser el fichero joined.msh, un mallado gmsh con 2 Physical Surfaces: 111 para la parte conducible y 222 para la no-conducible. Se puede leer más sobre gmsh en la sección VII. Se recomienda encarecidamente ver un videotutorial (sección XIII.1).

Una vez tenemos el fichero joined.msh en la carpeta s1_mesh\salida, podemos proseguir con los scripts, lanzando zgui(1) para el FATHER.


```
> cd c:\project\father
> zgui(1)
```



s4_terrain

Las opciones en el paso s4_terrain son:

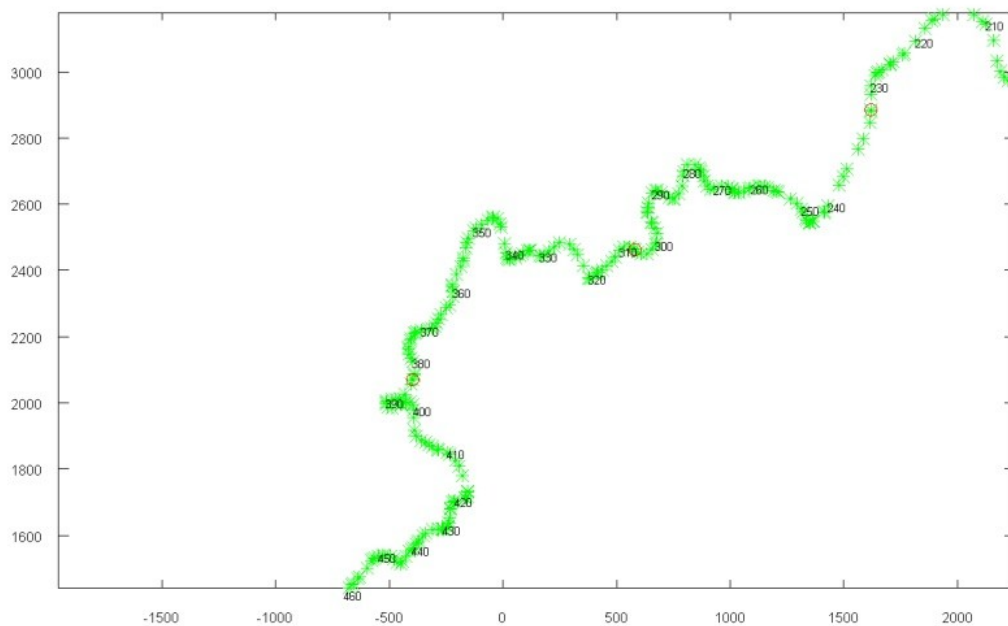
- 1) Crear sasplanet.hlg. Hay que crear este fichero si se quieren incluir imágenes de satélite en el proyecto. Este fichero está preparado para ser abierto con SASPLANET. Se puede leer más en la sección VIII.
- 2) Crear test.obj. Se crea un objeto 3D (formato .obj) en la carpeta s4_terrain\salida. Este objeto se puede emplear para comprobar la malla creada o quizá sea el resultado final que buscábamos de los scripts.

s7_walls

Se crean muros invisibles que impiden alcanzar la zona no conducible. Es opcional. Solo tiene sentido para RBR, no para rFactor.

s10_split

En primer lugar creamos puntos de ruptura (script split_track(n), siendo n el número de segmentos que queremos tener). Este botón crea un fichero llamado '**pos_nodes.txt**' que contiene los números de los nodos en los que la carretera cambia de segmento. También se muestra una gráfica que nos permite comprobar la situación de dichos nodos desde una vista en planta del trazado. Si queremos usar otros puntos de ruptura basta con **editar pos_nodes.txt** y luego pulsar en el siguiente botón "Dividir la carretera". El segundo botón (llama al script partir_track) usa pos_nodes.txt como entrada por lo que este fichero determina en última instancia cuántos segmentos tenemos (y dónde empiezan y acaban).



Una vez la carretera se ha troceado (tramos de 1Km parecen adecuados por razones de eficiencia), hay que trocear el terreno empleando una rejilla MxN grid. Si tenemos imágenes de fondo y las vamos a usar como textura para el terreno debemos emplear exactamente las mismas dimensiones MxN que hemos usado para trocear las imágenes. Se recomienda trocear por razones de eficiencia pero los usuarios de rFactor deben saber que demasiados polígonos (>1000) en un área de terreno puede hacer que BTB no consiga exportar el proyecto para rFactor.

Si se selecciona la opción “Mezclar con fondo”, las áreas de terreno tendrán las imágenes de satélite como textura. Leer sección VIII para ampliar.

s9_join

Si se desea tener imágenes de satélite como imágenes de fondo en BTB, hay que indicar el directorio correcto donde están los ficheros .dat (leer sección VIII para ampliar). En caso contrario hay que escribir un directorio que no exista. Si se selecciona la opción “Mezclar con fondo” en el paso s10_split SE DEBEN tener las imágenes de fondo, pues de lo contrario es imposible abrir el Venue.xml con BTB .

Finalmente se pueden añadir notas de copiloto a la ruta principal y crear el s9_join\salida\Venue.xml, el resultado final del proceso. Buscar la carpeta My Projects en el directorio de instalación del BTB. Crear una carpeta llamada NAME. Copiar Venue.xml dentro de NAME. Crear una carpeta dentro de NAME, llamada XPacks y copiar 3 ficheros dentro: default.zip y common.zip (copiar de otros proyectos de los que hay en My Projects) y WP.zip. También hay que añadir los archivos .dds si se usan imágenes de fondo (ver capítulo VIII).

IV.2 Proyectos One-track

Los proyectos que solo tienen una ruta son una versión simplificada de los multi-track: básicamente los mismos pasos pero no hay que procesar los SONS. En lugar de crear joined.msh a partir de joined.geo, crearemos anchors_carretera.msh a partir de anchors_carretera.geo.

Cuando se trabaja con una sola ruta deberíamos leer la palabra "NORMAL", en lugar de "FATHER"/"SON" en el GUI.

V TRABAJAR CON DIFERENTES FUENTES DE DATOS DE ELEVACIÓN

V.1 Google Earth

Es posible dar elevación a los ficheros gridXXX.kml usando un wrapper de Python para Google Earth COM API. También se pueden emplear BTBLofty o 3DRouteBuilder, but el proceso será más tedioso (y puede que más caro):

Hay varias opciones para dar elevación a los ficheros gridXXX.kml:

- 1) Comprar 3D Route Builder. A menos que se paguen 15 euros, el límite por fichero son 200 puntos
- 2) Usar BTBLofty. Funciona bien en Windows XP, pero no en Windows7. Aun así, en Windows XP hay que procesar los ficheros a mano: uno tras otro empleando una gran cantidad de tiempo.
- 3) Usar el script raise_kml. Hace lo mismo que las dos opciones anteriores. Funciona en Windows XP y Windows 7 y procesa automáticamente todos los ficheros gridXXX.kml.

Para usar raise_kml hay que realizar los siguientes pasos:

Instalar Python

1. Instalar Python 2.7 en el directorio [C:\Python27](http://python.org/ftp/python/2.7/python-2.7.msi). NO INSTALARLO EN UN DIRECTORIO DIFERENTE
<http://python.org/ftp/python/2.7/python-2.7.msi>
2. Instalar pywin32
<http://sourceforge.net/projects/pywin32/files/pywin32/Build216/pywin32-216.win32-py2.7.exe/download>
3. Instalar pygoogleeearth
<http://pypi.python.org/packages/any/p/pygoogleeearth/pygoogleeearth-0.0.2.win32.exe#md5=7e92b3cf1dcb4a5493aacb393a9e54a2>
4. Editar el fichero c:\Python27\lib\site-packages\pygoogleeearth\geapplication.py cambiando (línea 231)

```
return gehelper.point_dict_from_terrain_point(terrain_point)
```

por:

```
return terrain_point
```

V.2 AGR Data

Se aceptan ficheros ASCII Grid con formato:

```
NCOLS 601
NROWS 401
XLLCORNER 714000
YLLCORNER 4328800
CELLSIZE 25
NODATA_VALUE -999
33.381 33.279 33.102 32.982 32.809.....
```

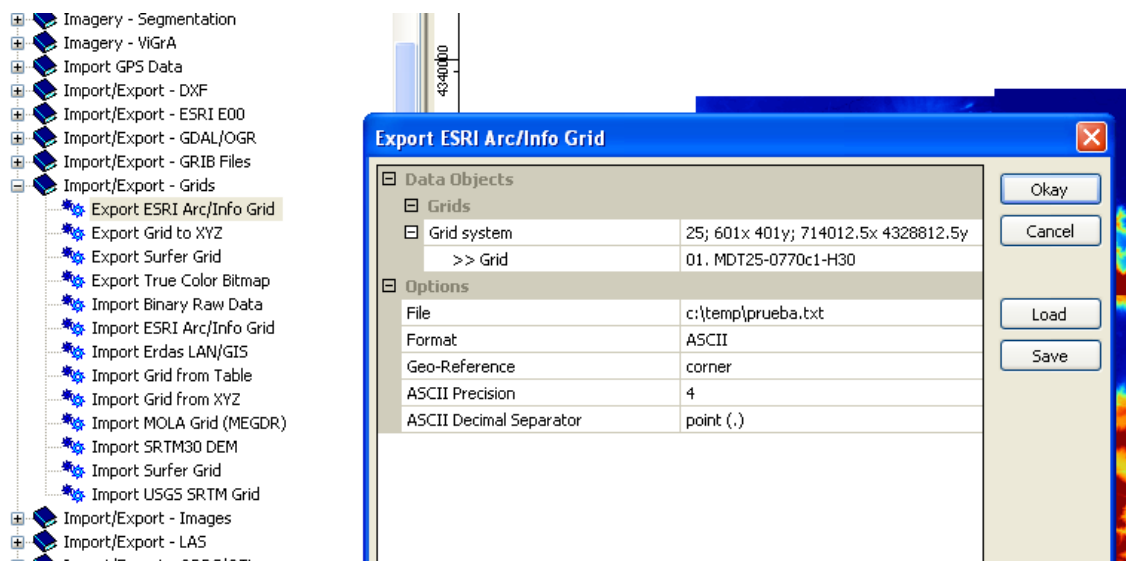
. Las coordenadas deben ser **UTM** (*Universal Transverse Mercator*).

Uso:

- 4) Crear una carpeta llamada “agr” en la misma carpeta en la que está "project" (o las carpetas del father's y los sons en los proyectos multi-track). Ejemplo: si trabajamos en un proyecto one-track en c:\project, crearemos la carpeta c:\agr.
- 5) Copia los ficheros .agr a la carpeta creada en el paso 1). Renombra esos ficheros con extensión .agr si es necesario.
- 6) Usa los scripts con normalidad. Siguiendo los pasos recomendados en la pantalla el paso s2_elevation hay que saltárselo, pues ya tenemos datos de altura.

NOTE: si se tienen datos con un formato diferente se puede emplear software libre para convertirlos (como SAGA-GIS, donde se importa la rejilla y se selecciona el *Module Import/Export - Grids \ Export ESRI Arc/Info Grid*, usando formato ASCII y corner geo-reference).

<http://sourceforge.net/projects/saga-gis/>



NOTA: para España se pueden descargar de forma gratuita datos .agr con 5 o 25m de resolución. Basta crear un usuario y descargar datos MDT05 o MDT25.

<http://centrodedescargas.cnig.es/CentroDescargas/>

De la siguiente gráfica se puede sacar el código adecuado a la zona que nos interesa:

http://centrodedescargas.cnig.es/CentroDescargas/equipamiento/cuadrícula_MTN50.png

Búsqueda Avanzada

Seleccione Productos

Modelo Digital del Terreno - MDT25

[Ver descripción de los productos](#)

Seleccione División administrativa:

División administrativa

Seleccione Hoja del MTN50:

70

[Ver cuadrícula del MTN50](#)

[Ver lista de Productos](#)

Resultados de su búsqueda / [Volver a buscar](#)

1

página 1 de 1

| Producto | Archivo | Formato | Tamaño(MB) | Descargar |
|------------------------------------|------------------|---------|------------|---------------------------|
| Modelo Digital del Terreno - MDT25 | MDT25-0070c4.zip | AGR | 1,32 | Descargar |
| Modelo Digital del Terreno - MDT25 | MDT25-0070c1.zip | AGR | 1,32 | Descargar |
| Modelo Digital del Terreno - MDT25 | MDT25-0070c2.zip | AGR | 1,35 | Descargar |
| Modelo Digital del Terreno - MDT25 | MDT25-0070c3.zip | AGR | 1,30 | Descargar |

página 1 de 1

NOTE: para Cataluña se pueden descargar datos con 15m de resolución de:

http://www.icc.cat/vissir2/?lang=es_ES

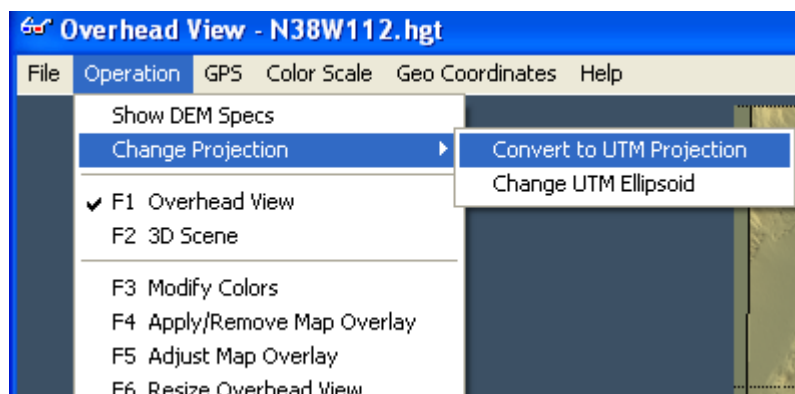
cliqueando en el mapa y seleccionando "otras". El formato es exactamente el mismo que usado por el cnig, pero la resolución es diferente. La extensión de los ficheros debe cambiarse de ".txt" a ".agr".

NOTA: Los ficheros MDT05 pueden ser demasiado grandes para usarlos directamente con los scripts. El script `split_agr` puede usarse para trocear un fichero MDT05 en ficheros más pequeños. Por ejemplo, para trocear **MDT05-0667-H30.ASC** y crear 5x5 ficheros (con extensión .AGR):

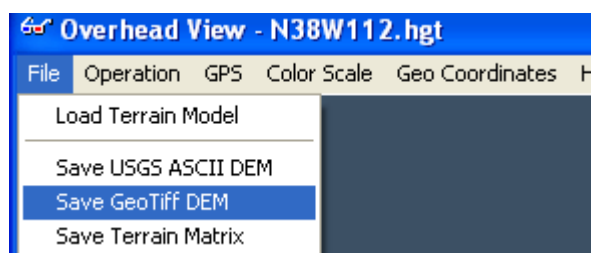
```
split_agr('MDT05-0667-H30',5)
```

NOTA: los ficheros .hgt se pueden abrir con 3dem para proyectarlos a UTM (Operation\Change Projection)

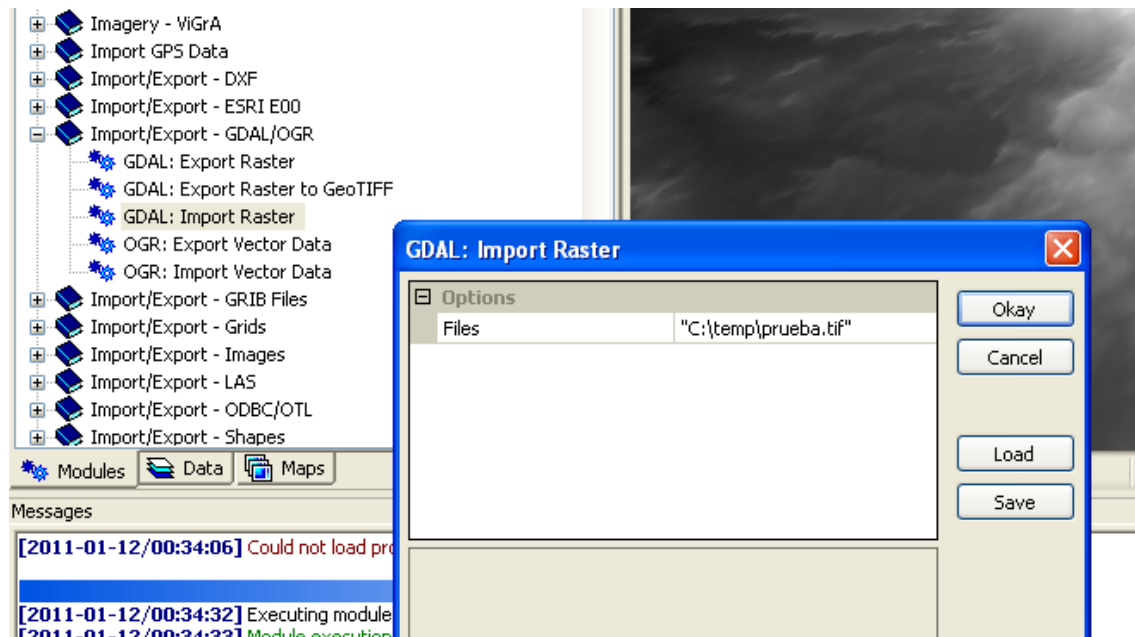
<http://www.viewfinderpanoramas.org/3dem.zip>



Luego se guardan con formato Geotiff dem:



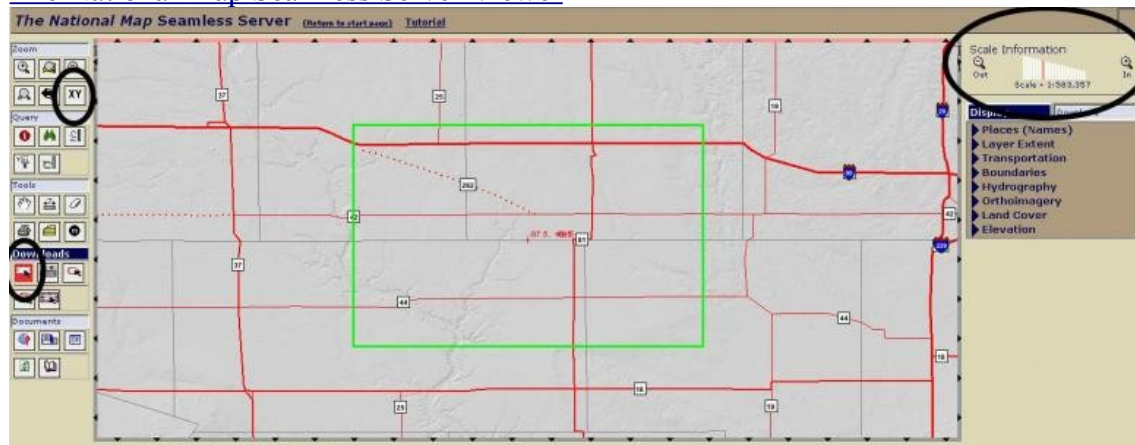
Y se importan con SAGA GIS:



V.3 Seamless server

Esta web ofrece datos cada 10 o 30m para USA y 90m (y a menudo con algunos puntos sin altura) para el resto del mundo.

The National Map Seamless Server Viewer



Para comprobar el tipo de datos que ofrecen para nuestra zona de interés seleccionamos la opción **Zoom->XY** y escribimos las coordenadas de un punto. Luego hacemos zoom para ampliar (parte superior derecha de la pantalla). Con el **Downloads->Rectangle** enmarcamos la zona de interés. Si el rectángulo es demasiado grande, se pondrá rojo. En la ventana pop-up clickamos en el botón **Modify Data Request**. Buscamos la sección **Elevation** y seleccionamos la fuente de datos. Por

ejemplo "National Elevation Dataset (NED) 1/3 Arc Second" y elegimos formato "**GRIDFLOAT**". Clickamos en el botón "Save Changes & Return to Summary" y si no hay ningún problema el botón "**Download**" aparecerá ante nosotros. Es una forma sencilla y rápida de obtener buenos datos de elevación. Si no nos interesa un tramo de USA siempre podemos recurrir a GE a través de 3DRoute Builder, BTBLofly o raise_kml.

Nota:

3 arcosecond \sim 90m

1 arcosecond \sim 30m

1/3 arcosecond \sim 10m

Los datos en formato gridfloat se pueden usar para crear lamalla.mat mediante el script **leer_gridfloat**. Este script no puede usarse hasta que se haya completado el paso s0_import.

Ejemplo:

```
> cd ../s2_elevation
```

```
> leer_gridfloat("file.hdr","file.flt")
```

V.4 Ficheros hgt

Ejemplo:

```
> cd ../s2_elevation
```

```
> lee_hgt("N41W009.hgt",[41 42],[-9 -8])
```

Los ficheros .hgt pueden transformarse en lamalla.mat usando el script **lee_hgt**. Este script no puede usarse hasta que el paso s0_import se haya completado.

VI AJUSTE FINO DEL PERFIL EN ALTURA DE LA CARRETERA

Hay dos acciones avanzadas relacionadas con el perfil en altura de la carretera:

- 1) Changing by hand the profile proposed by the scripts
- 2) Cambiar el terreno para que se adapte a la carretera que hemos creado

VI.1 Retocar a mano el perfil

Hay un tutorial en javascript: <http://www.mediafire.com/?pdrdg6q9kp5zv3x>

Ejecutar dar_altura con normalidad

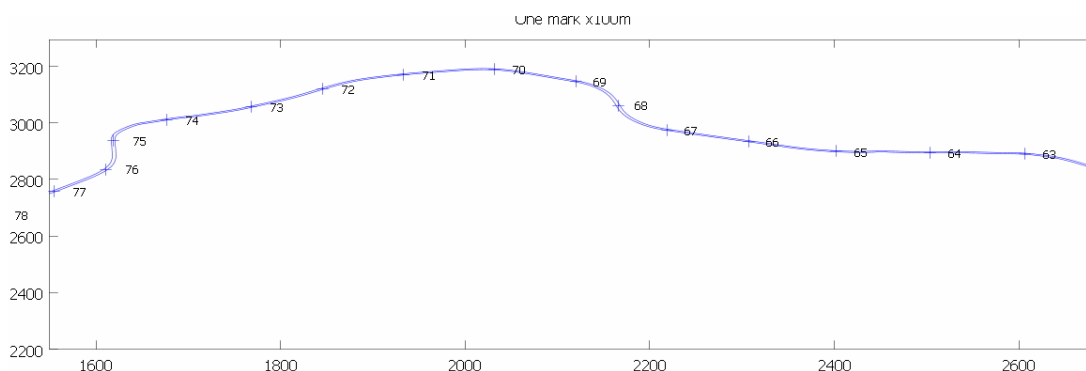
```
> dar_altura(15,0.3,-0.3)
```

```
> 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
```

Ahora **dar_altura** espera a que el usuario teclee “n” o “e” y luego <ENTER>

- “n” si se quiere modificar un tramo del perfil en altura
- “e” si hemos terminado la edición

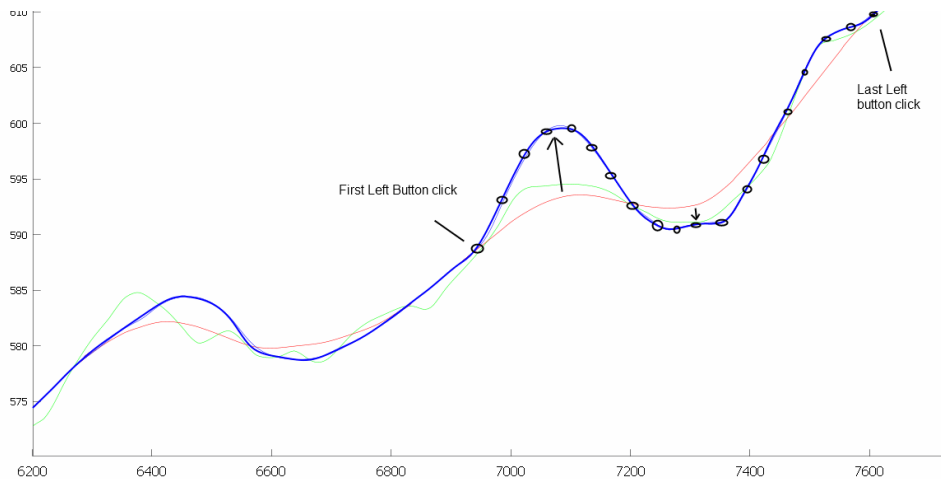
Pero **antes de pulsar “n”, debemos ampliar la zona de interés**. Y antes de ampliar probablemente necesitaremos echarle un vistazo a la otra gráfica creada por dar_altura. Muestra la distancia recorrida en cada punto de la ruta (multiplicar por 100 para tener la distancia en metros)



Si por ejemplo quieres cambiar el perfil en altura entre las posiciones 6800m y 7600m, hay que hacer un zoom de esa zona (se selecciona con el botón derecho) en la otra figura. Tras hacer el zoom se puede pulsar “n” en la ventana de texto de octave. El mensaje en la ventana de texto debería cambiar:

```
Leyendo el fichero retoques.txt
Cerrando el fichero
      e_ end          n_ new segment
n
Left click to add point. Right click to finish
```

Ahora debemos hacer click en la gráfica de elevación con el botón izquierdo del ratón. **De izquierda a derecha**. Se puede usar cualquier número de puntos (≥ 3). Para acabar basta con hacer click en cualquier parte de la ventana con el botón derecho del ratón.



Se nos vuelve a pedir que escojamos “e” o “n”.

Si pulsamos “e” el script dar_altura termina su labor. Dentro de “s3_road\salida” se puede encontrar un fichero “Venue.xml” que se podría abrir con BTB para comprobar la forma de la carretera.

Importante: Todos los retoques manuales se graban en un fichero llamado “retoques.txt”. Es un fichero de texto que podría editarse a mano llegado el momento. Cada línea del fichero se compone por el número de puntos de un segmento y las coordenadas (x,y) coordinates de dichos puntos.

Cada vez que se llama a dar_altura ‘retoques.txt’ es leído y su información se aplica al perfil. Los nuevos cambios se añaden al final de ‘retoques.txt’.

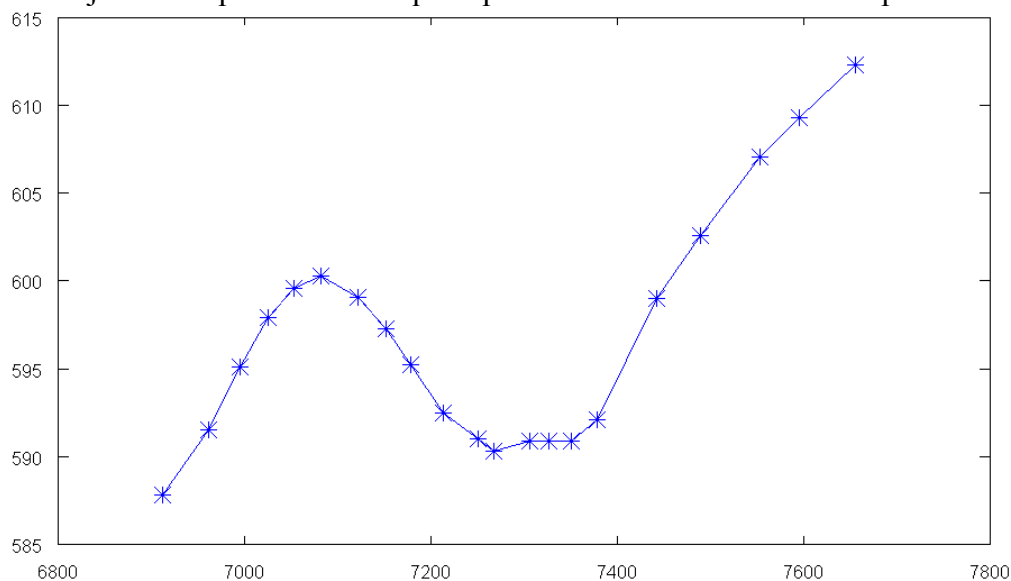
Si se quieren borrar los cambios hechos a mano hay que borrar el fichero “retoques.txt”.

Los cambios contenidos en retoques.txt se aplican en el mismo orden en que aparecen en el fichero.

Ejemplo de línea contenida en retoques.txt:

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

Si dibujamos los puntos vemos que representan uno de los cambios aplicados:

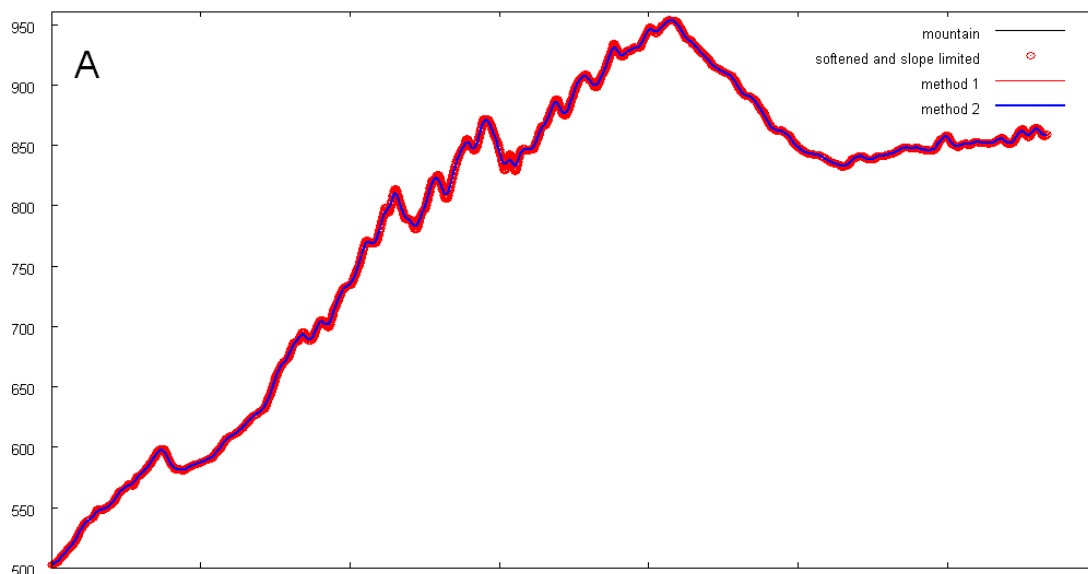


El primer parámetro de dar_altura es un factor de suavizado. Debemos escoger un valor que minimice el número de ediciones manuales requeridas. Siempre debe ser impar.

El perfil final se creará con un spline sacado de un dato de elevación cada 25m. Eso significa que *las pequeñas irregularidades creadas al editar a mano NO son importantes*. Esa distancia (25m por defecto cuando no se usa el GUI) se puede cambiar usando un 4º parámetro en el script dar_altura.

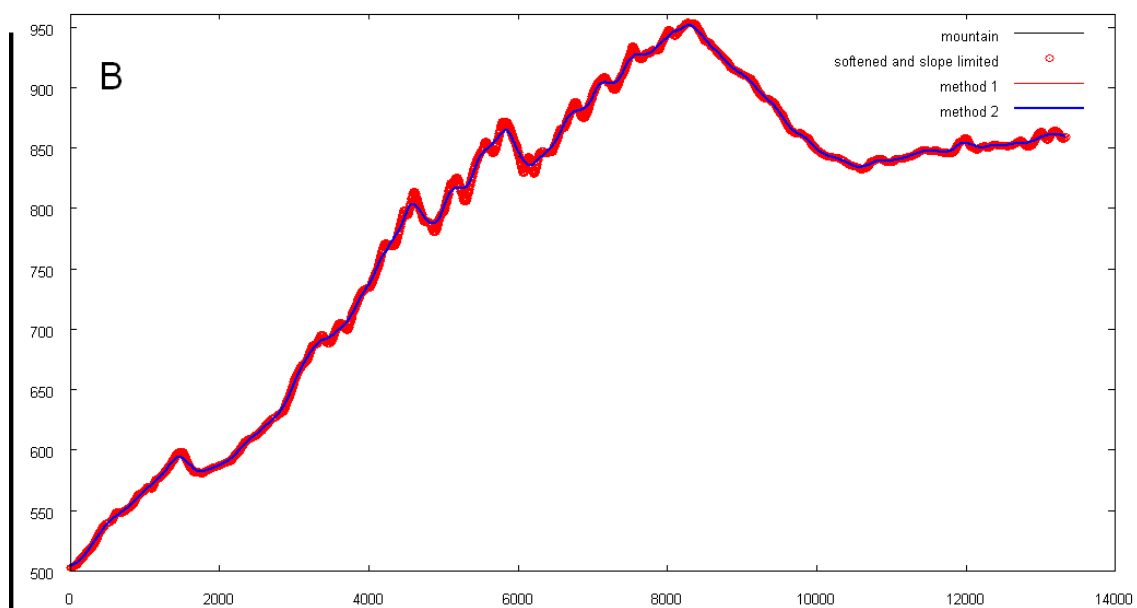
VI.2 Uso del script “corregir”

Supongamos que queremos crear un tramo pero al ejecutar “dar_altura” comprobamos que los datos de altura no son nada buenos y que la carretera sube y baja sin sentido (ver Fig A). Los cambios de elevación se producen obviamente por malos datos de altura (podríamos acudir a Google Earth para confirmarlo).

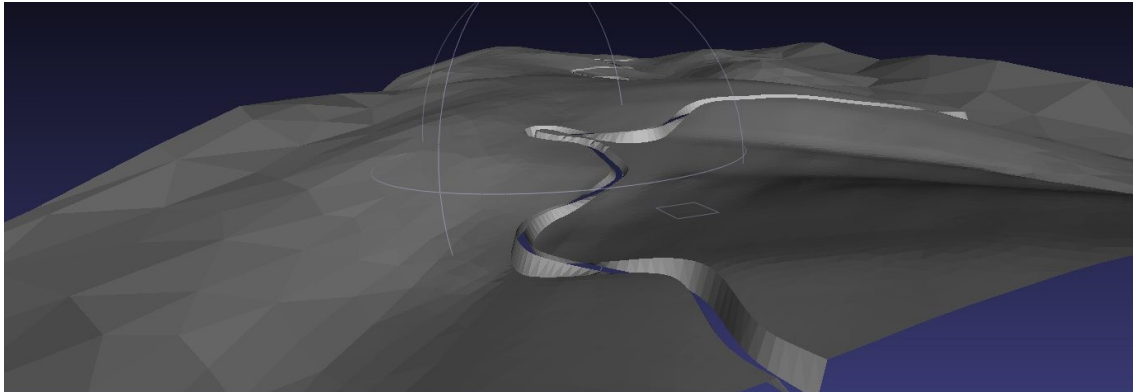


El script “corregir” puede cambiar los datos de altura de la siguiente forma. En primer lugar creamos un perfil suave para la carretera con `dar_altura` (ver Fig B).

`> dar_altura(53,1,-1,100)`



Esta carretera no se ajusta bien a los datos de altura que tenemos. Si siguiésemos con el proceso usando esta carretera, obtendríamos algo como esto:



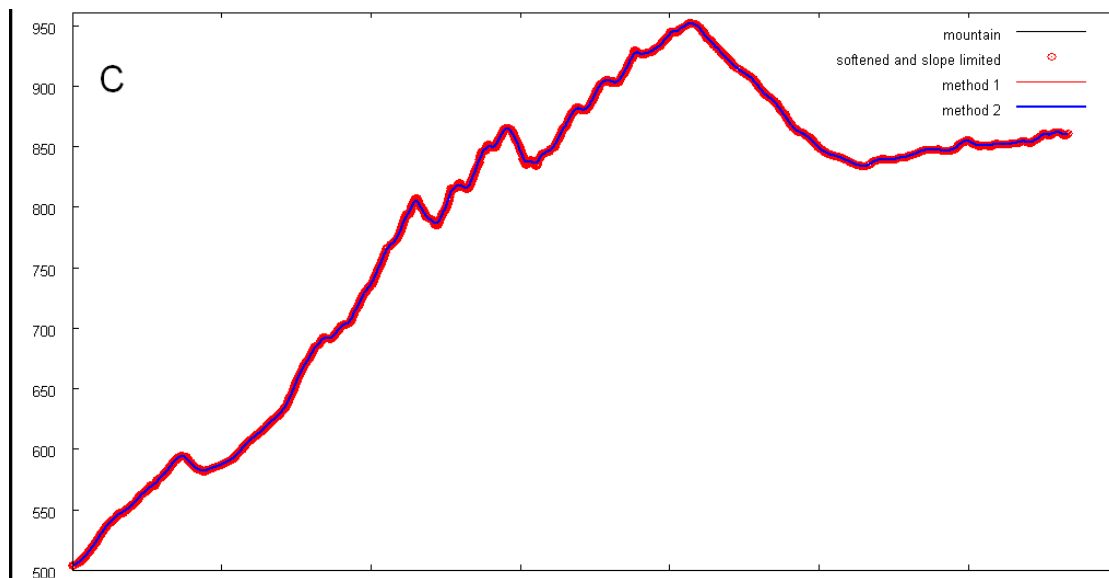
Ahora queremos modificar los datos de altura para que se adapten a nuestra deseada "carretera suave":

```
> corregir
```

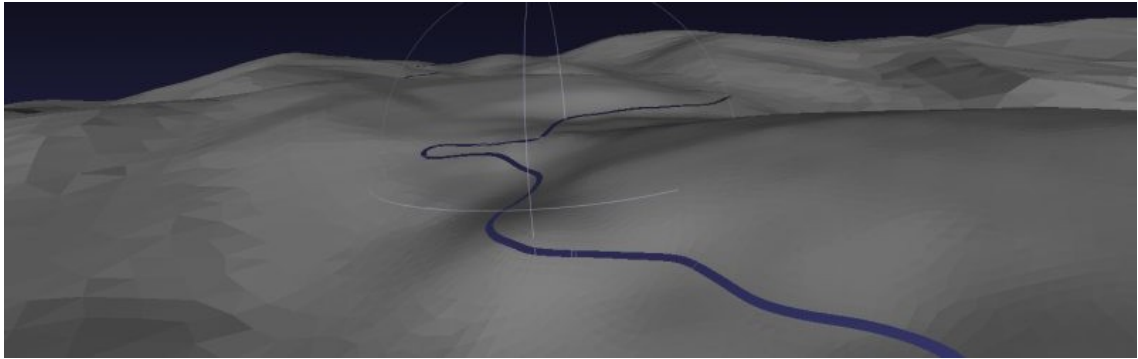
Y ejecutamos de nuevo dar_altura para crear la carretera final, una carretera adaptada a los nuevos datos de elevación:

```
> dar_altura(21,0.25,-0.25)
```

Se puede ver (Fig C) que ahora la carretera (curva azul) es prácticamente tan suave como la deseada, y la montaña (círculos rojos) no tiene grandes diferencias en elevación con dicha carretera. Se pueden comparar los círculos rojos y la curva azul en Fig B y Fig C. La carretera es muy parecida, **pero “corregir” ha cambiado los datos de elevación para adaptarlos a la carretera.**



Una vez tenemos una carretera con un perfil en altura que nos guste, podemos seguir con los siguientes scripts. El resultado puede ser tan bueno como:



Scripts que se han ejecutado (los parámetros podrían haber sido diferentes):

```
cd ../s3_road
s3_coge_datos
creartrack1
dar_altura(53,1,-1,100)
corregir
dar_altura(21,0.25,-0.25)
```

Si se quiere comenzar el proceso desde el principio basta con ejecutar de nuevo `s3_coge_datos` en `s3_road`, y tendremos los datos de elevación originales listos para ejecutar `creartrack1` y el resto de los scripts. Si no se quiere usar “corregir”, basta con usar `dar_altura` con normalidad.

VII USANDO GMSH

VII.1 ¿Qué son `anchors_carretera.geo` y `joined.geo`?

`anchors_carretera.geo` y `joined.geo` son ficheros que deben ser procesados con gmsh. Contienen:

- `TerrainAnchors`, los puntos en que carretera y terreno se unen. Estos puntos están unidos con rectas.
- Puntos auxiliares, a ambos lados de la carretera y a una distancia configurable como parámetro en el diseño.
- “Plane surfaces” ya creadas para la zona conducible.

El usuario tiene que definir la superficie no conducible. Ésta es la única parte con cierta dificultad del “método zaxxon”, pero hay varios videotutoriales que muestran los pasos a seguir.

VII.2 Conceptos basicos de gmsh

Si dos puntos de la frontera de una malla están unidos con una recta, ambos puntos formarán parte del mallado.

Si dos puntos de la frontera de una malla están unidos con un spline, esos puntos no necesariamente formarán parte del mallado. gmsh buscará la posición óptima de los nodos de los triángulos en esa frontera.

Cuando se trabaja con gmsh cada punto tiene una "**characteristic length**", que es la longitud deseada para los lados de los triángulos en ese punto. Por ejemplo, en el fichero `anchors_carretera.geo` (es un fichero de texto), los `Anchors` de carretera tienen una `characteristic length` llamada "`cl1`", con un valor de 20m. No obstante los `Anchor` están unidos con rectas por lo que los triángulos en contacto con la carretera tendrán un lado definido por la posición de los `Anchors` (típicamente separados 5m). Por defecto los scripts fijan el tamaño de los triángulos mediante un mecanismo denominado `thresholds` y no empleando `characteristic lengths` (ver sección VII.8).

VII.3 Controles básicos de gmsh

- Botón derecho del ratón y arrastrar: mover el proyecto
- Ctrl + botón izquierdo del ratón y arrastrar: hacer zoom de una zona
- Clickear 1:1 en la esquina inferior izquierda: zoom ajustado
- Clickear Z en la esquina inferior izquierda: restaurar vista desde arriba
- Botón izquierdo del ratón y arrastrar: rotación 3D
- Rueda del ratón: zoom

VII.4 ¿Cuáles son los límites de la zona no conducible?

La zona no conducible no debería exceder los datos de elevación disponibles. De otro modo scripts como procesar_nodostxt acabarán con errores. Si no has seleccionado en el GUI la opción de ver los límites de los datos de elevación disponibles y deseas verlos en la pantalla de gmsb, ejecuta addgrid en la carpeta s1_mesh:

> **addgrid(1,1)**

Donde los dos parámetros son el número de divisiones en horizontal y vertical de la rejilla que se está añadiendo. Este comando solo funcionará si se están usando ficheros lamalla.mat files, como es el caso cuando se sacan datos de altura de Google Earth, pero no si se usan datos AGR. Abre el .geo de nuevo para ver los cambios.

NO se deben alcanzar los límites de los datos de elevación con los mallados hechos en gmsb. De otro modos los scripts fallarán al no encontrar datos de altura para algunos de los puntos del mallado.

VII.5 ¿Cómo debo crear la frontera externa del terreno no conducible?

Para poder mallar primero hay que crear las “Plane surfaces” que mallar. Para crear las plane surfaces primero hay que definir sus límites y para definir sus límites necesitamos antes definir algunos puntos.

Seleccionamos:

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

Una vez el razón está en la posición deseada para el punto mantenemos pulsado “*shift*” en el teclado y las coordenadas de congelan. Llevamos el puntero del ratón hasta la ventana "Contextual Geometry Definitions" y allí ponemos la coordenada Z=0 y la Characteristic Length a c13 c13 (c13 es un valor definido en la parte inicial de anchors_carretera.geo). Clickeamos en añadir.

Una vez definidos los puntos que vamos a usar, seleccionamos:

Geometry-> Spline

y creamos un spline seleccionando los puntos por los que debe pasar. Una vez cerrada la curva podemos definir la “Plane surface” que tiene el spline como su borde externo y la superficie conducible como su hueco interno.

Geometry -> Plane Surface

Clickeamos en la frontera externa hasta que esté toda roja. Si se seleccionado por completo gmsh nos pedirá que seleccionemos el borde del hueco interno ("Select hole boundaries"). Debemos seleccionar todos los segmentos del borde externo de la zona conducible, hasta que todos ellos tengan color rojo. En ese momento la opción deshacer (“undo”) desaparecerá de la parte superior de la ventana de gmsh. Pulsamos “e” para crear la “Plane surface”.

Podemos mallear para comprobar que todo es correcto hasta ahora:

Mesh -> 2D

y todas las plane surfaces ya definidas serán malladas.

Vamos a tener dos tipos de superficies:

- Conducibles: entre los puntos auxiliares y la carretera
- No conducibles: entre la frontera externa y los puntos auxiliares.

Podemos usar la opción Tools->Statistics para averiguar cuántos triángulos tendrá nuestro diseño.

VII.6 ¿Cuál es la salida del proceso?

Si salvamos el mallado, un fichero llamado joined.msh/anchors_carretera.msh será creado. Este fichero contiene dos tipos de líneas:

1) Nodos. Por ejemplo el nodo #14:

14 2149.778 5113.46 0

2) Triángulos (gmsh les llama elements, y BTB faces). Por ejemplo el triángulo 21222

pertenece a la physical surface de los conducibles (111) y está definido por tres nodos: 1490, 17130 and 1491:

```
21222 2 3 111 31672 0 1490 17130 1491
```

VII.7 ¿Cómo se definen las Physical Surface?

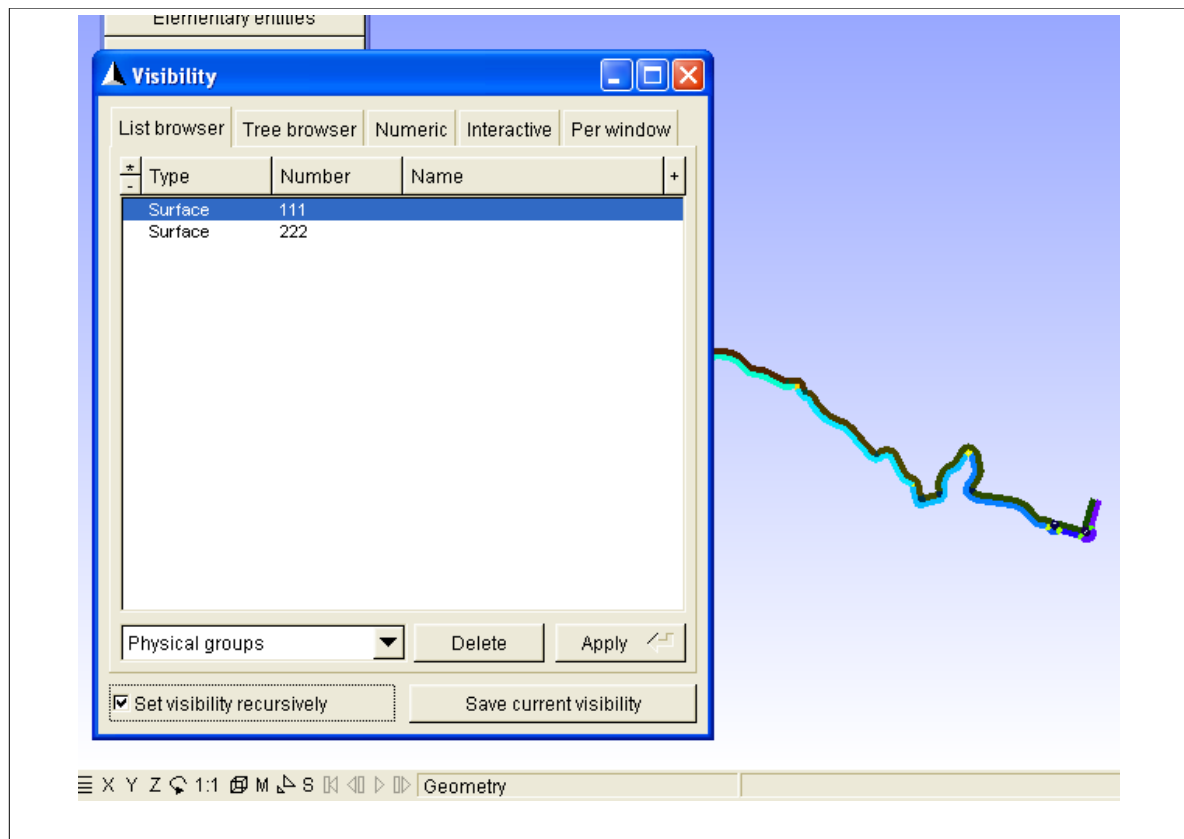
Tenemos que crear 2 physical surfaces, conductor (numerada 111) y no conductor (numerada 222). Cada una de ellas se define como la lista de “plane surfaces” que pertenecen a esa physical surface.

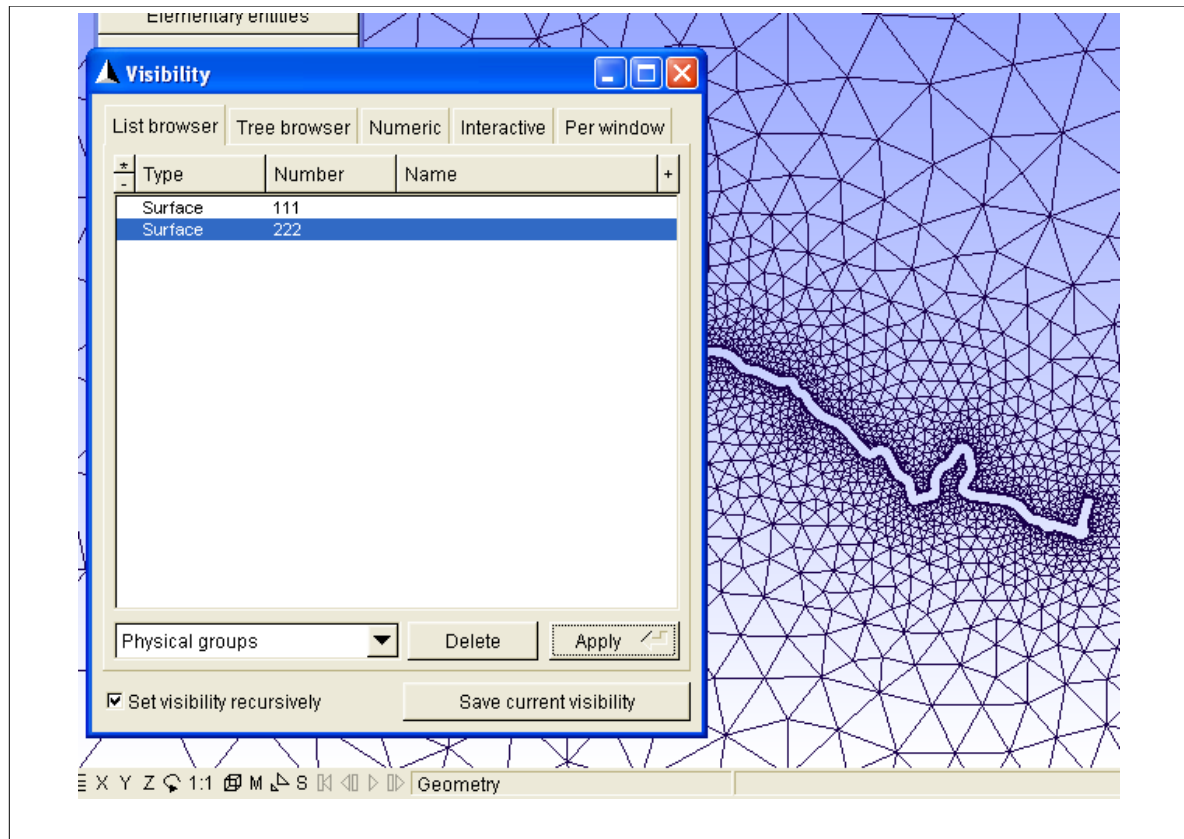
La lista inicial de plane surfaces para la Physical Surface 111 es el contenido del fichero salida\phys111.txt (la lista está ahí desde que los scripts permitieron los proyectos multitrack). Si tu proyecto no tiene rutas adicionales, simplemente añade el contenido de phys111.txt al final de anchors_carretera.geo:

“Physical Surface(111)= {contents of phys111.txt};”.

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).

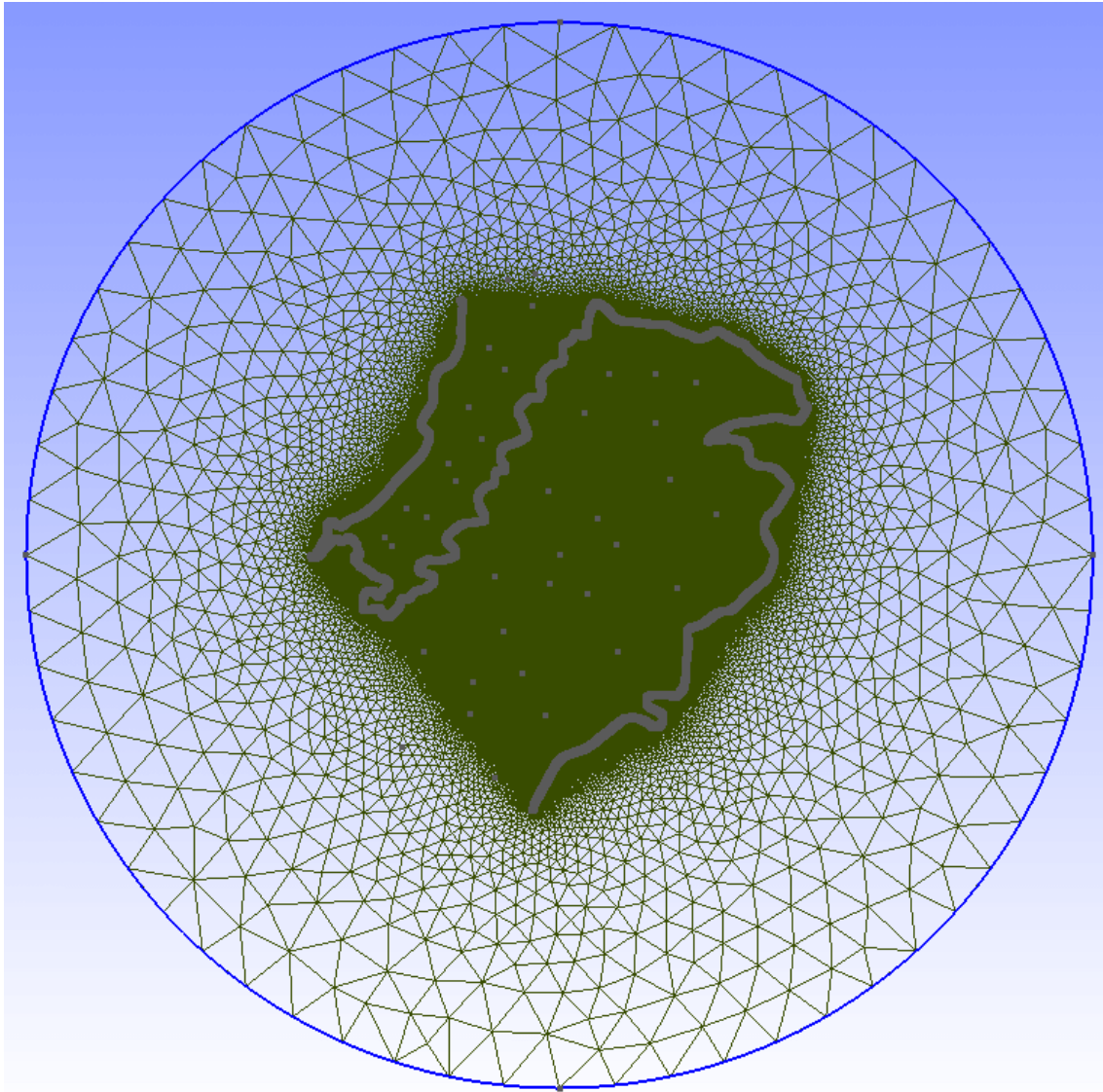
Only the plane surfaces that belong to a physical surface will be part of the .msh file, so it is important to pay attention when defining the physical surfaces. **Once you have finished meshing it is recommended to open anchors_carretera.msh/joined.msh with gmsh to check that all the meshes are there.** Using the Visibility tool (and there Physical Groups in the List Browser tab) may help you as you can select what physical surface you want to see.



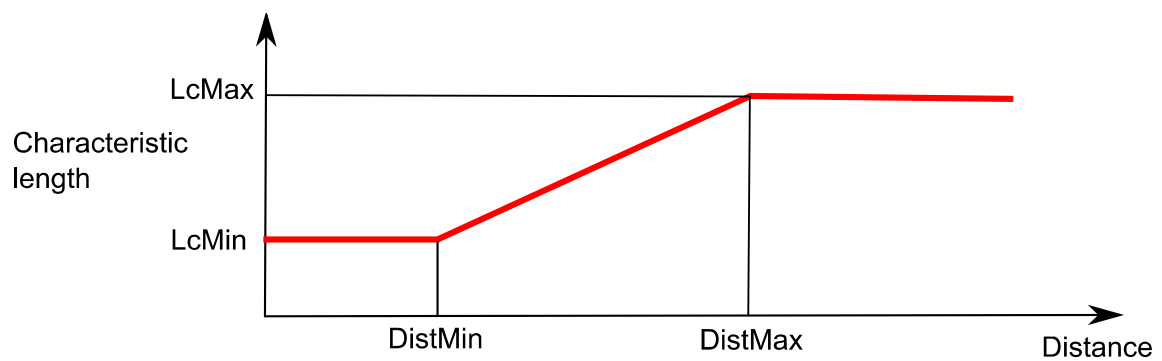


VII.8 Gmsh thresholds

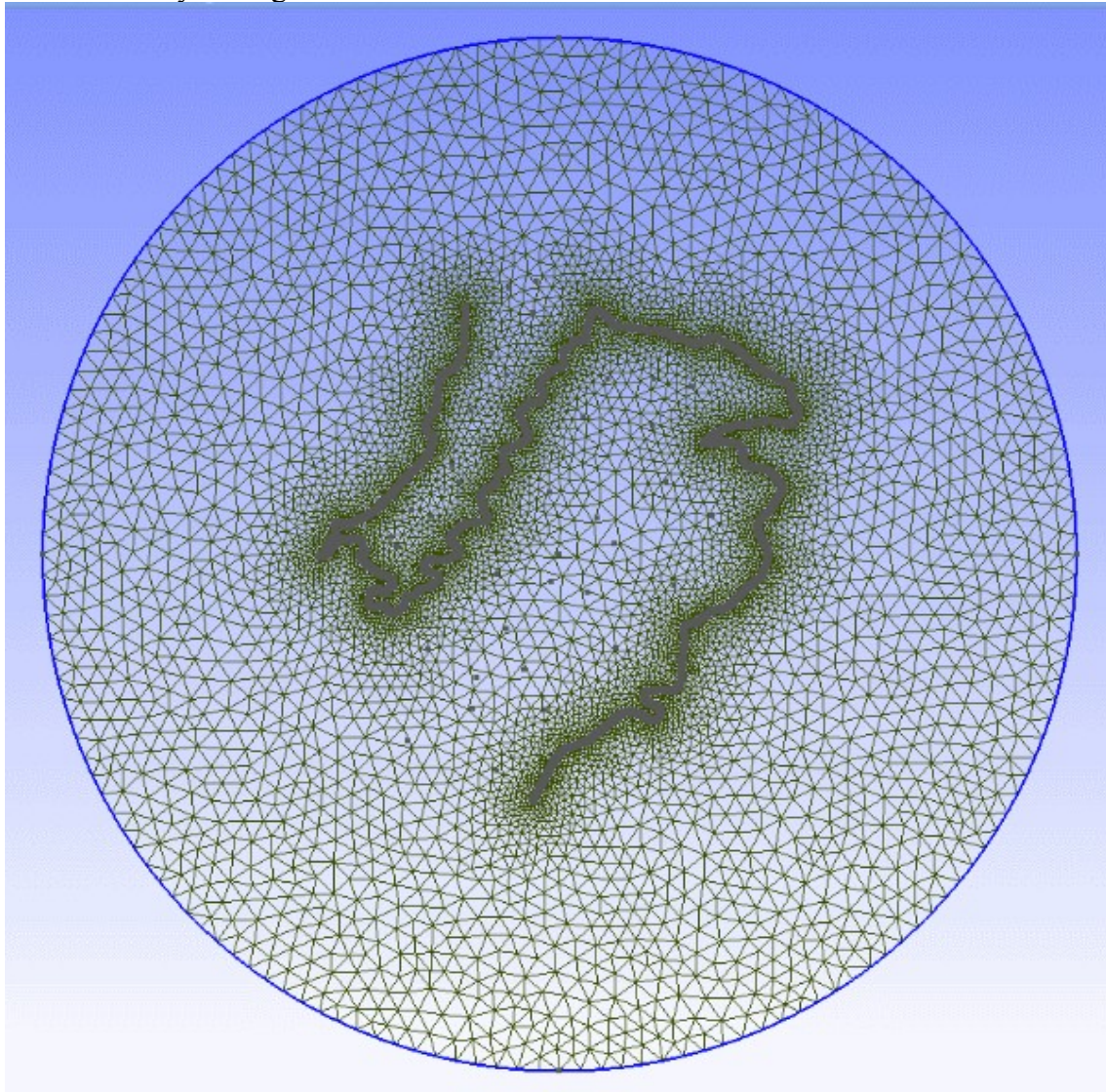
One way of controlling the triangles size with gmsh is giving each point a characteristic length (4th parameter in their definition). The problem with characteristic lengths is that gmsh creates too many triangles for U-shaped tracks (or segments of a track). For example:



Another way of controlling triangles' size is using *threshold fields*. For example you can ask gmsh to assign each triangle a size that is a function of the distance to the track.



The result may be as good as follows:



Gmsh code for using a threshold field is:

```

Field[1] = Attractor;
Field[1].NodesList = {1:3342}; //List of point of the track

Field[2] = Threshold;
Field[2].IField = 1;
Field[2].LcMin = 20;
Field[2].LcMax = 2000;
Field[2].DistMin = 1;
Field[2].DistMax = 10000;
Field[2].StopAtDistMax = 0;

Mesh.CharacteristicLengthExtendFromBoundary = 0;

Background Field=2;

```

mallado_regular or join_geos scripts include (in anchors_carretera.geo or joined.geo) the code needed for using the threshold fields. It is **activated by default**. If you don't want to activate the thresholds, just open the .geo file with a text editor and search for line **If (1)** and change it to **If (0)**.

You can only activate one threshold field unless you use a Min Field (read the notes at the end of this document). The threshold that is included automatically uses all the points defined so far in the .geo file, so all the tracks are considered for that threshold in multitrack projects.

The default parameters in the geo are set to (read s1_mesh\thresholds.txt):
 LcMin = 20; //(Characteristic length at a distance LcMin and below)
 LcMax = 2000; //(Characteristic length at a distance LcMax and above)
 DistMin = 1;
 DistMax = 10000;

NOTE:

It is also possible to change the threshold parameters within gmsh:

Select **Mesh > Define > Fields**.

In the Fields window click **Threshold**.

For example we reduce element sizes approaching the boundary limits:

1) Enter LcMax = 1000

2) Apply

3) Click 1D meshing.

Note: You should 1D before 2D each time new Threshold parameters are entered.

4) Click 2D meshing.

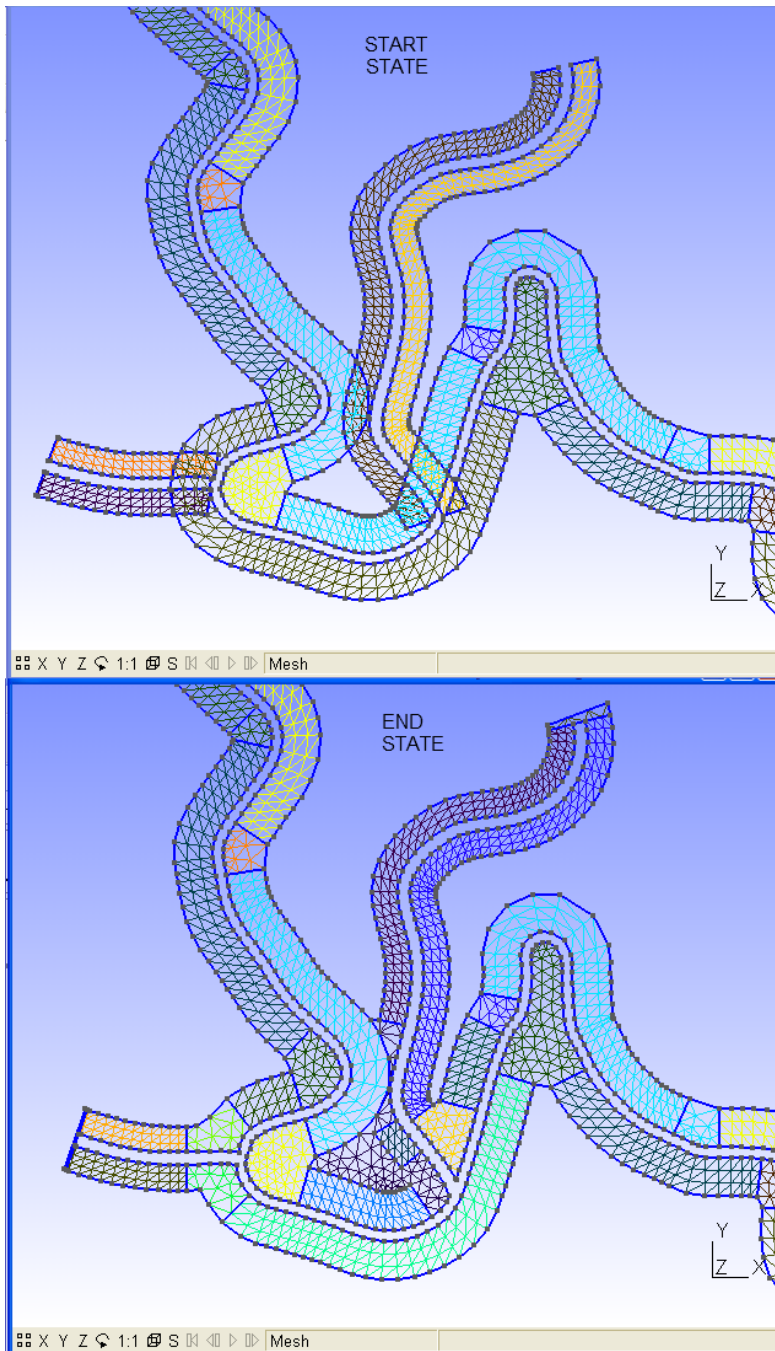
VII.9 Processing joined.geo with gmsh

When we open joined.geo with gmsh, we will find that the driveable meshes of the sons overlap the father's mesh. That must be fixed by hand, using gmsh and a text editor.

The steps we will follow are:

- 1) Fix overlapping of driveable zones
- 2) Create Physical Surface(111)
- 3) Create non-driveable zone and Physical Surface(222)
- 4) Final check

1) Fix overlapping



We have three options for fixing overlapping: 0) the fast, 1) the easy, and 2) the not-so-easy ways.

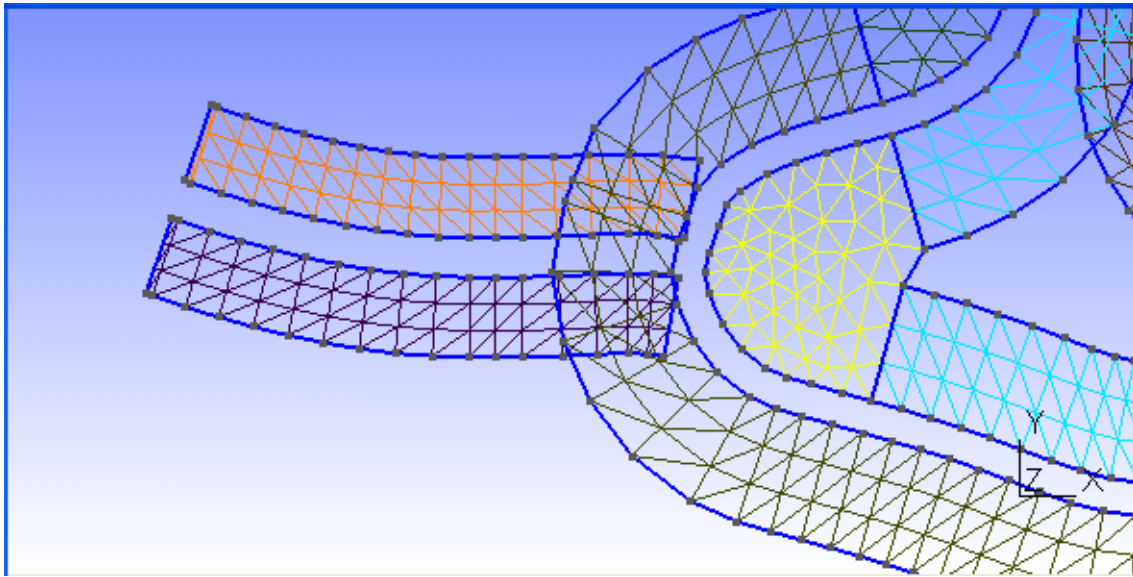
FAST WAY

I think this is the best option:

<http://btbtracks-rbr.foroactivo.com/t511-gmsh-editing#3345>

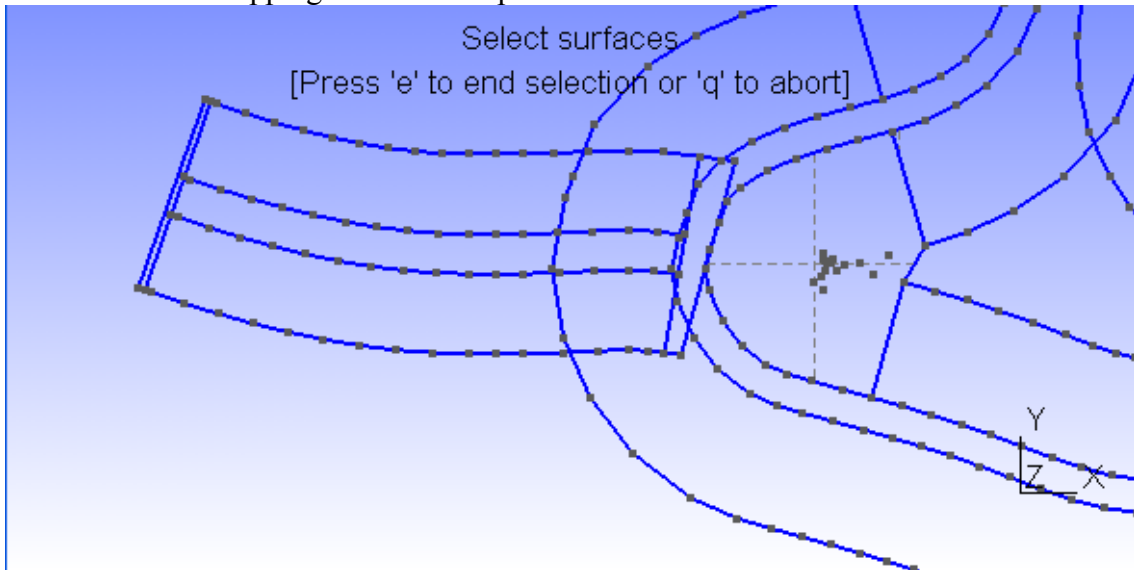
EASY WAY

- Delete overlapping surfaces
- Delete unnecessary lines (just for having a clear view)
- Create new surfaces



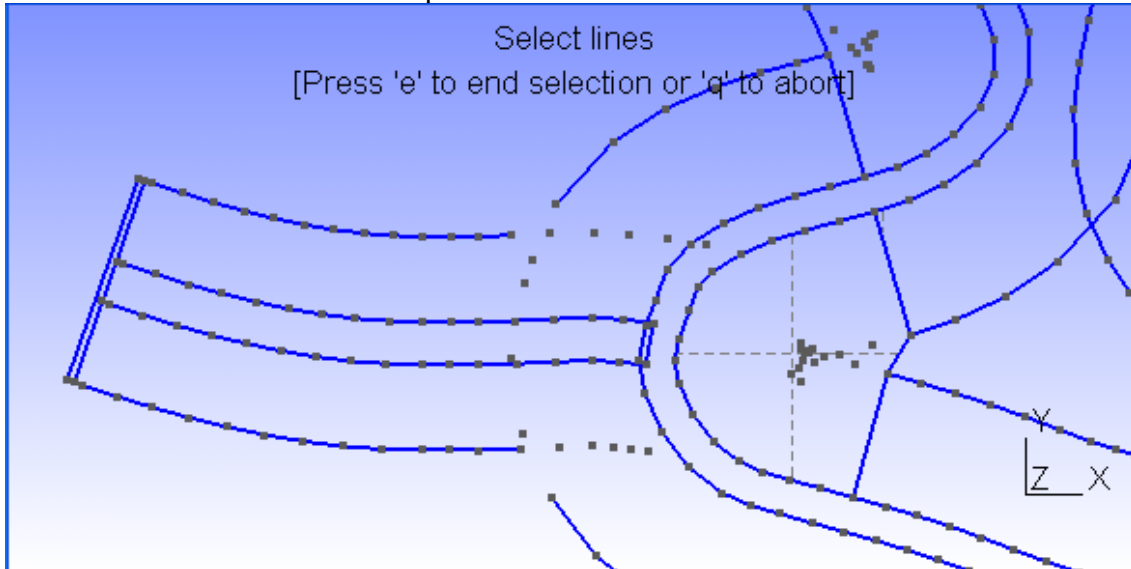
Geometry\Elementary entities\Delete\Surface.

Click on the overlapping surfaces and press “e” to delete



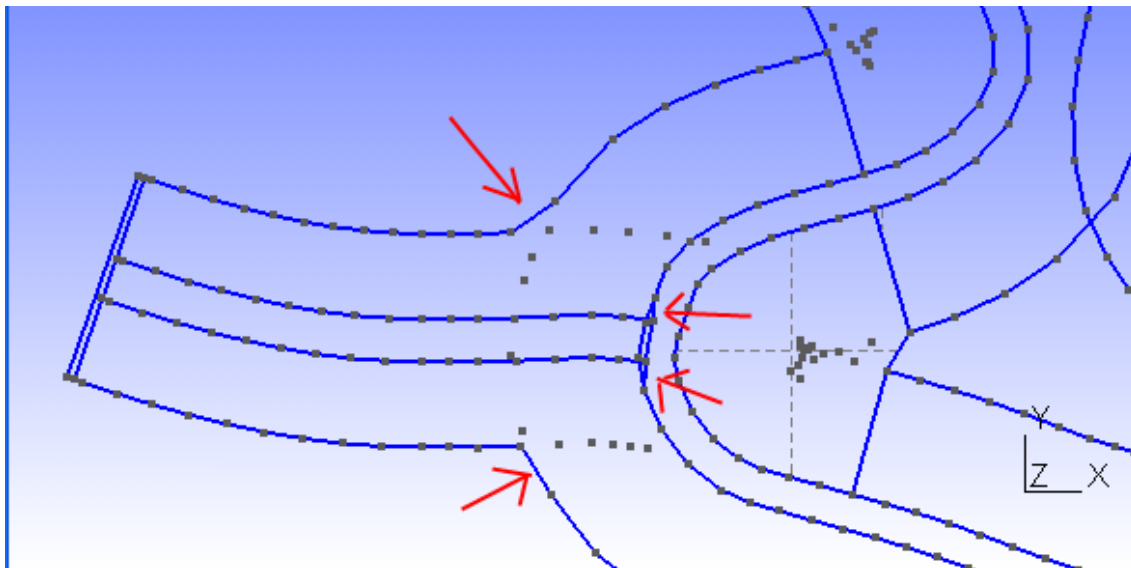
Geometry\Elementary entities\Delete\Line

Click on the unwanted lines and press “e” to delete



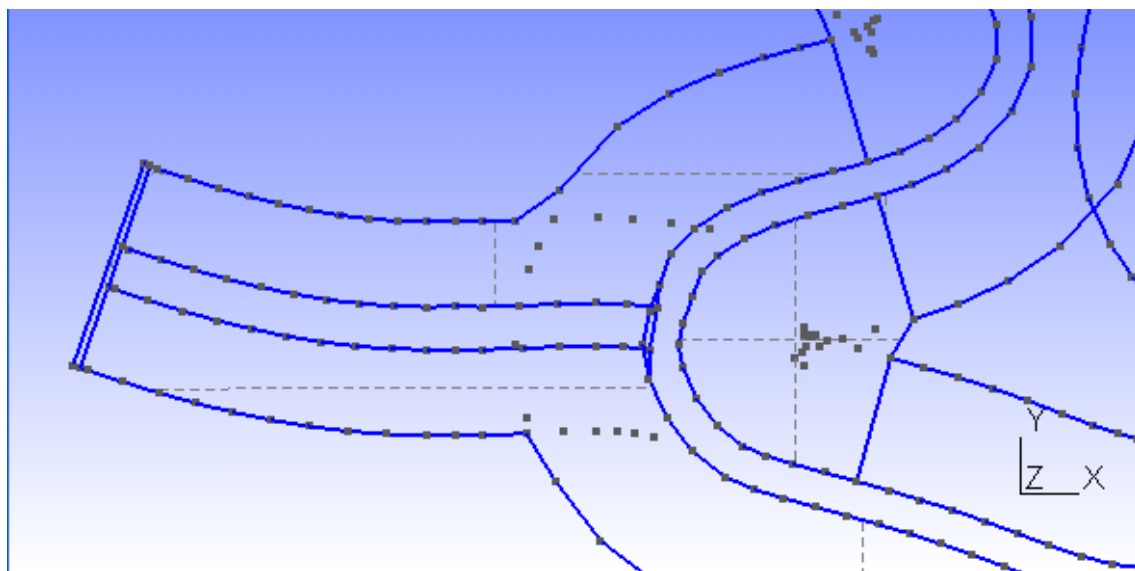
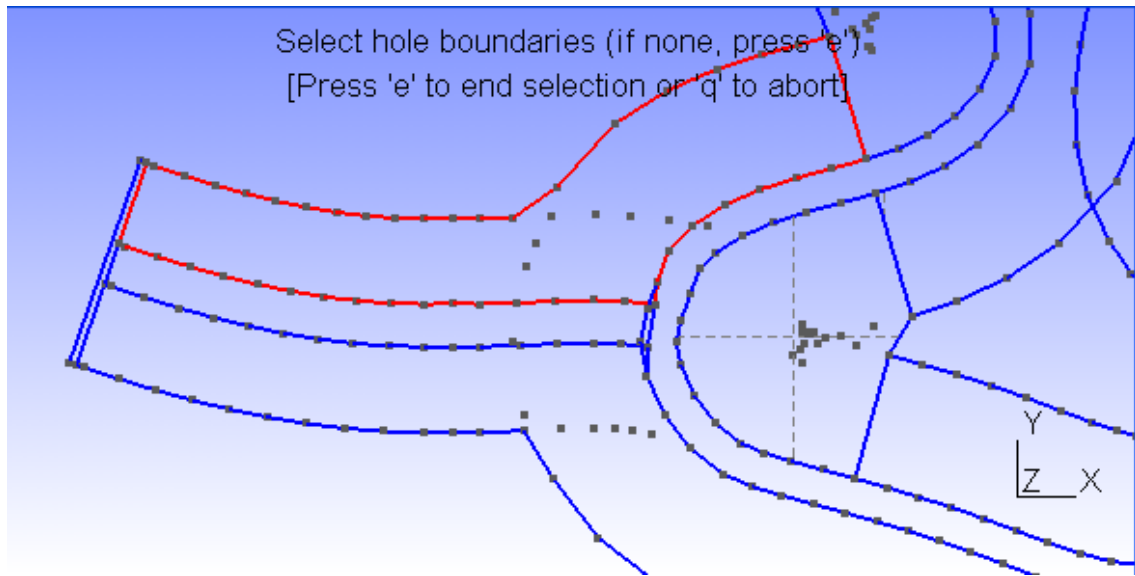
Geometry\Elementary entities\Add\New\Straight Line

Create the lines needed to close the surfaces

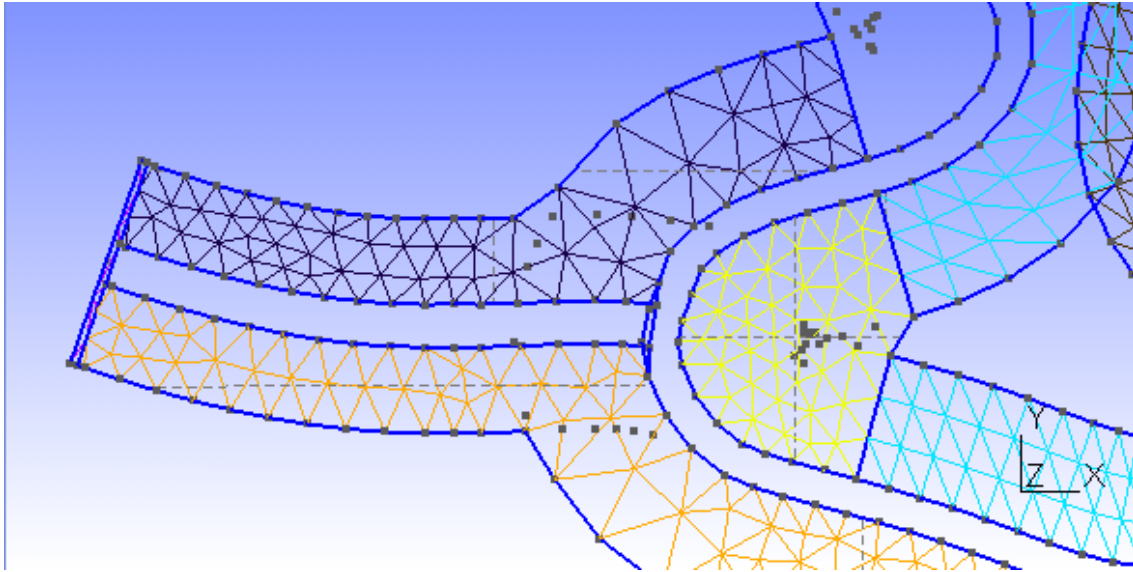


Geometry\Elementary entities\Add\New\Surface

Select the boundary of a surface and press “e”. When all the segments of the boundary are selected, option “u” disappears from screen top message. Repeat for all the surfaces needed.



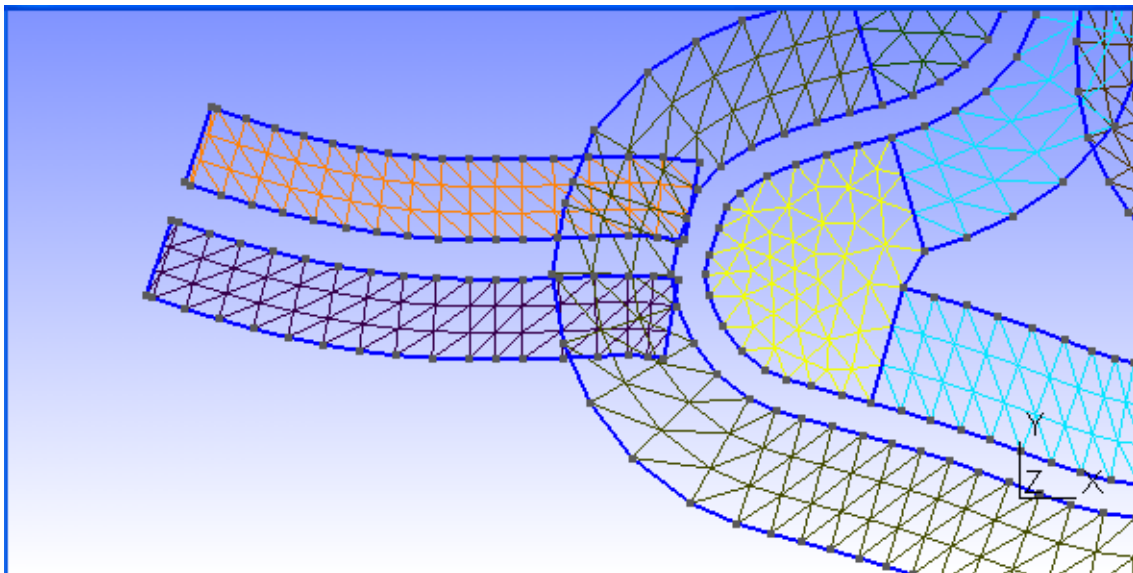
Mesh\2D
To check the resulting mesh



As you can see, with the “easy-way” we lose the regular pattern of the driveable mesh. The not-so-easy way tries to preserve that pattern, with a little extra effort.

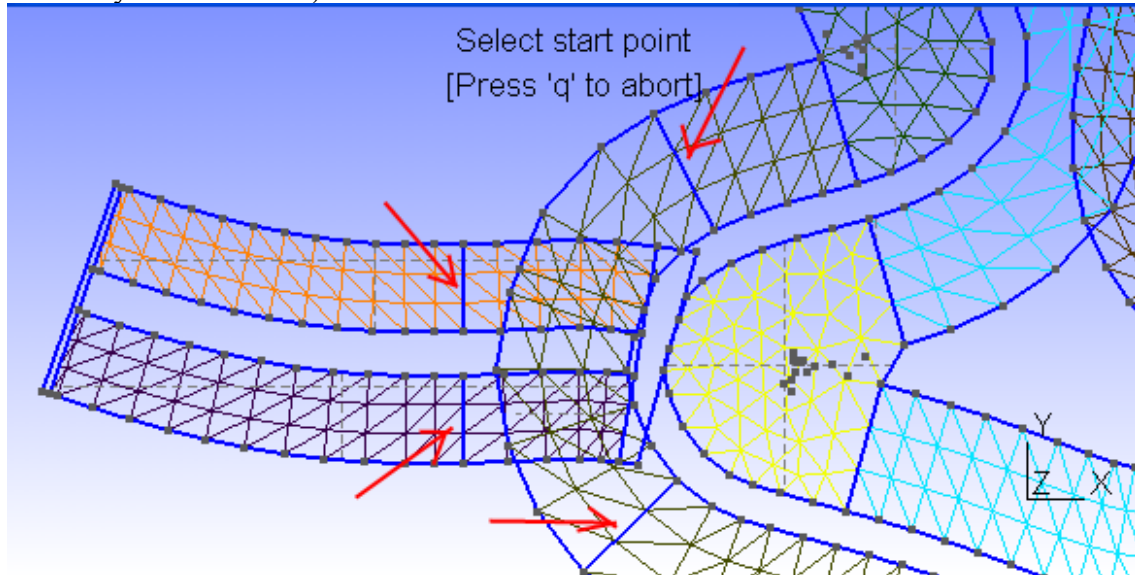
NOT-SO-EASY WAY

- Delete overlapping surfaces
- Delete unnecessary lines (just for having a clear view)
- Create regular surfaces
- Create irregular surfaces



Geometry\Elementary entities\Add\New\Straight Line

Create the transversal lines where the regular pattern will stop (first point on road, second one away from the road).



Open joined.geo with a text editor and declare the created lines as transfinite. You can use a for loop to make this step faster (copy-paste the loop and change starting and end line numbers).

```
Line(99992) = {97703, 97742}; AM
Line(99993) = {97685, 97724}; AM
Line(99994) = {1853, 31853}; AM
Line(99995) = {1845, 31845}; AM

For h In {99992:99995}
  Transfinite Line(h) = 4 Using Progression 1;
EndFor
```

gmsh has written this

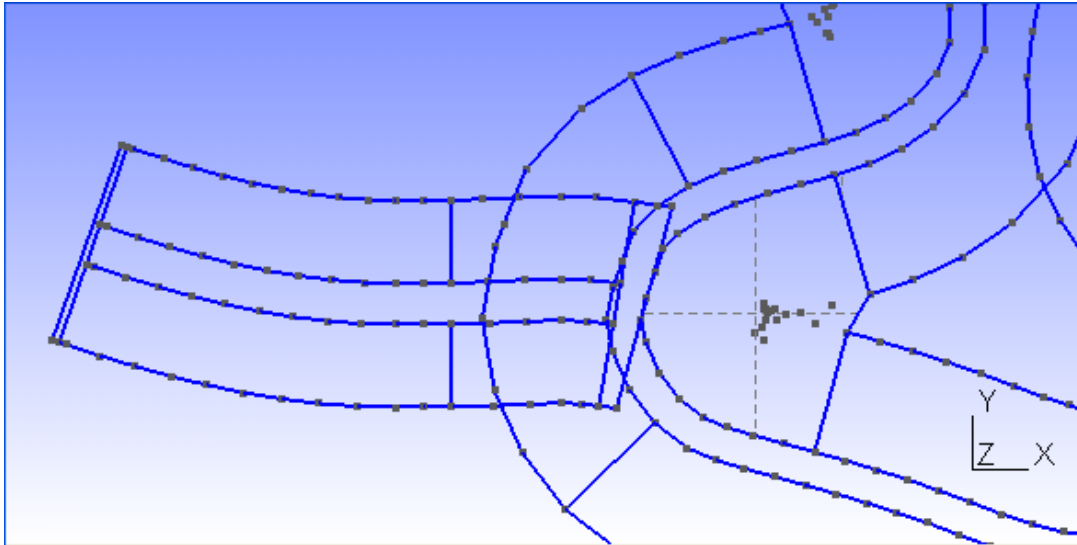
we add this

3 panels

In this example the 4 lines created had 3 panels, so they shared the “for loop”. Create this lines in groups of the same number of panels, so they can share the for loop. If you apply a wrong number of panels to a line, gmsh will refuse to create a mesh with a regular pattern for that surface.

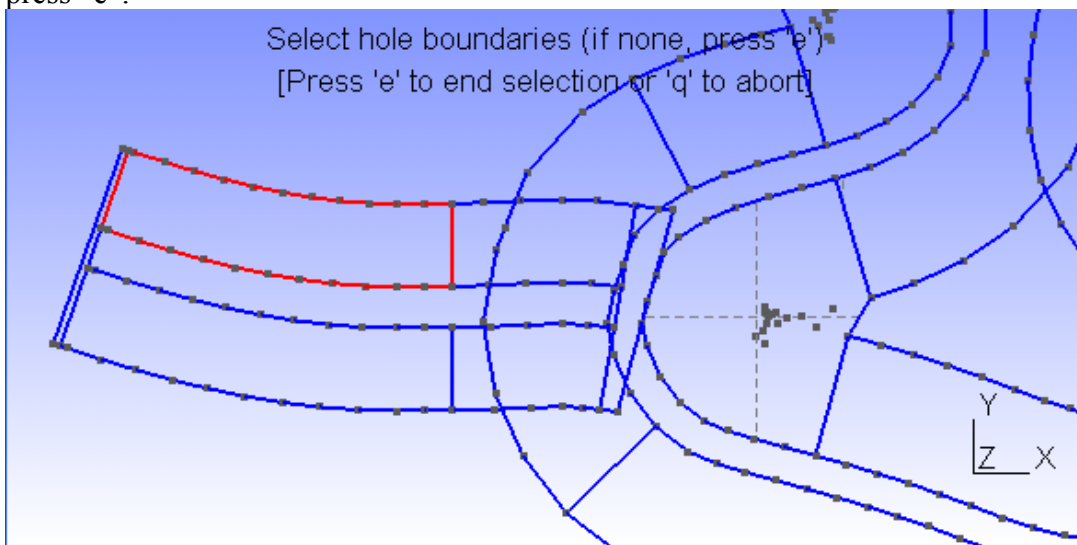
Geometry\Elementary entities\Delete\Surface.

Click on the overlapping surfaces and press “e” to delete



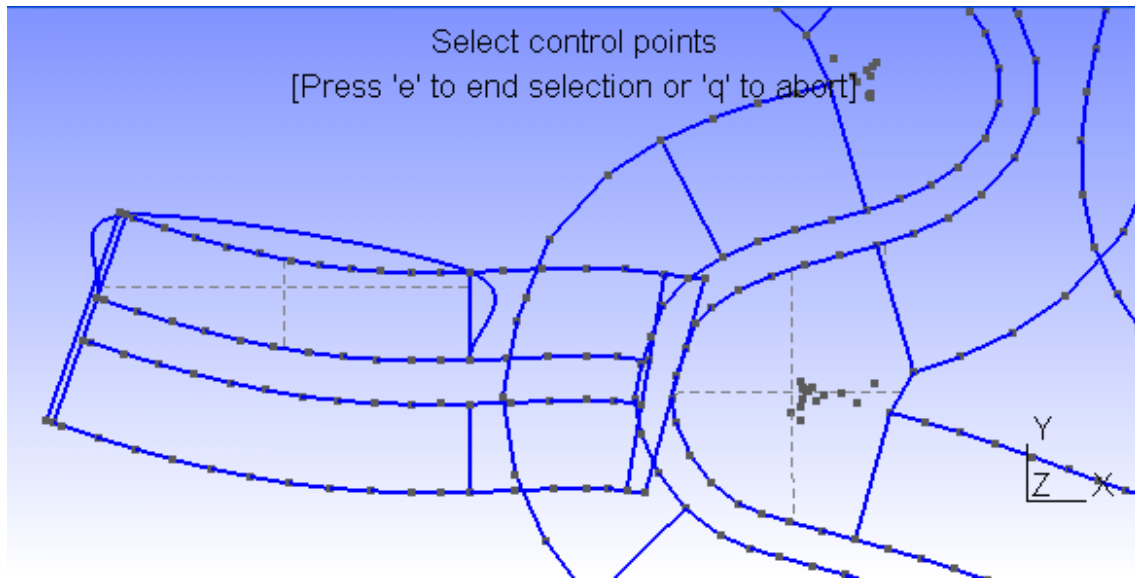
Geometry\Elementary entities\Add\New\Plane Surface

Select the boundary of one of the surfaces with regular pattern that need to be created, and press “e”.



Geometry\Elementary entities\Add\New\Spline

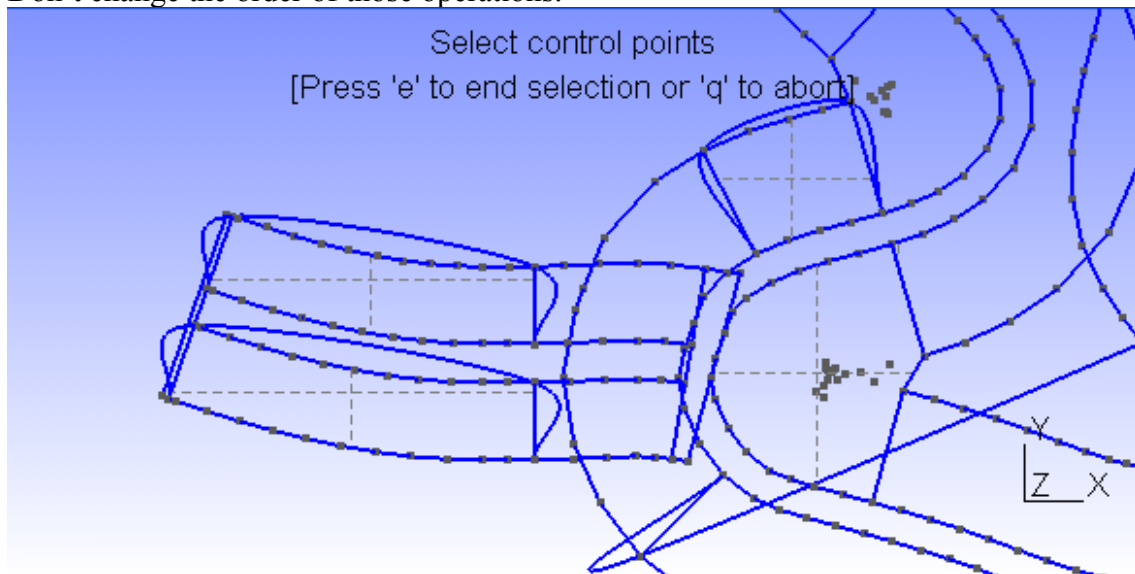
Create a spline using the four corners of the surface you have just created, and press “e”



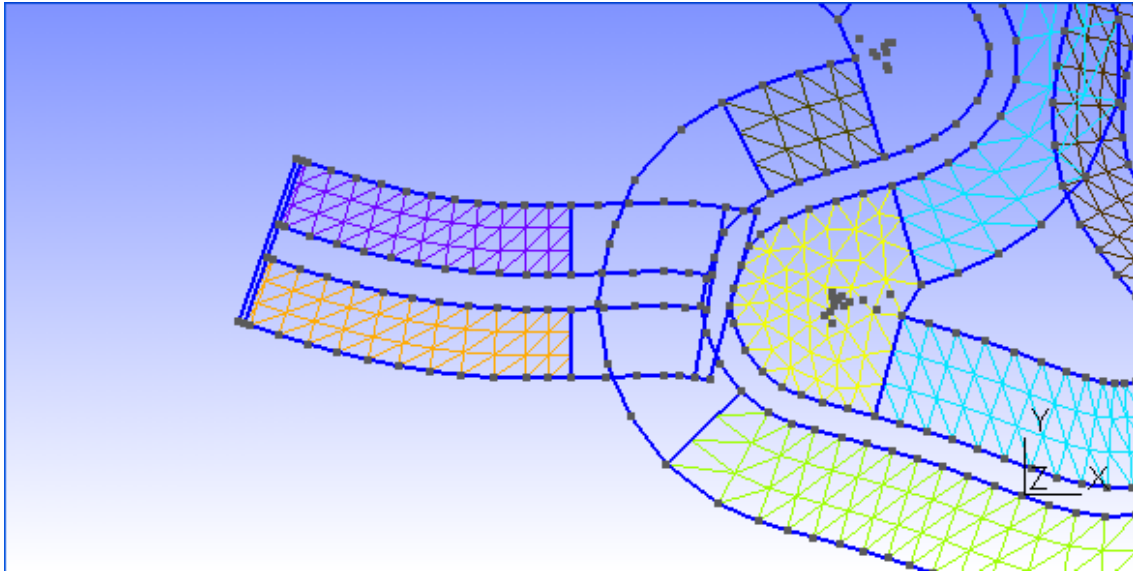
Now run script “addt” inside s1_mesh directory

```
octave-3.2.3.exe:7:f:\vmdata\tramos\tudons\tudons_father\s1_mesh
> addt
Leyendo el fichero salida\joined.geo
Cerrando el fichero
Escribiendo el fichero salida\joined.geo
Cerrando el fichero
octave-3.2.3.exe:8:f:\vmdata\tramos\tudons\tudons_father\s1_mesh
>
```

Repeat the procedure for all the regular surfaces: create surface, create spline, run “addt”. Don’t change the order of those operations.



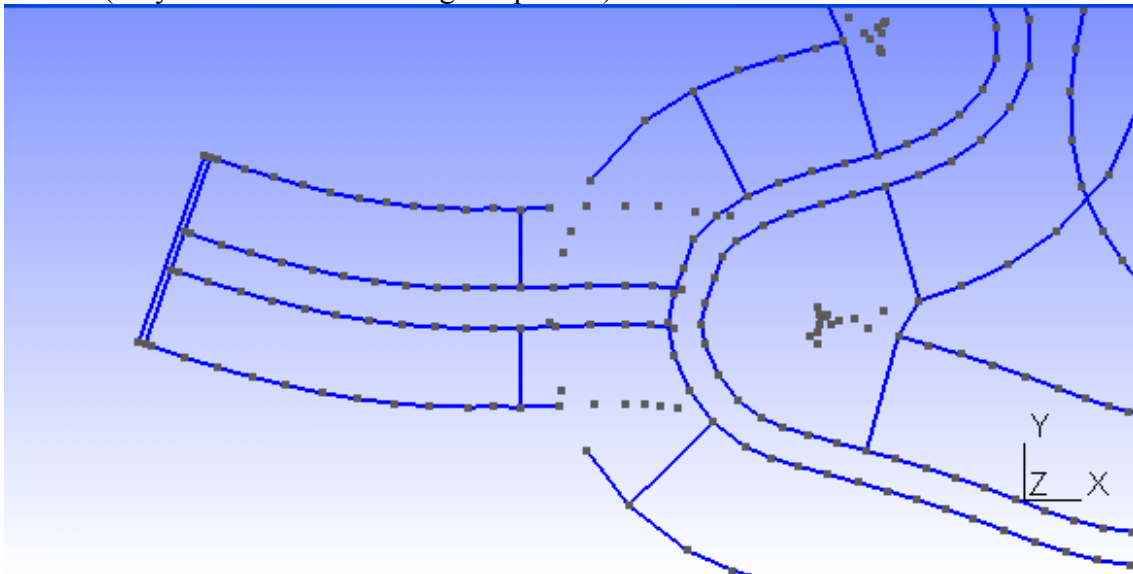
Now close gmsh, open again joined.geo with gmsh and “Mesh\2D”

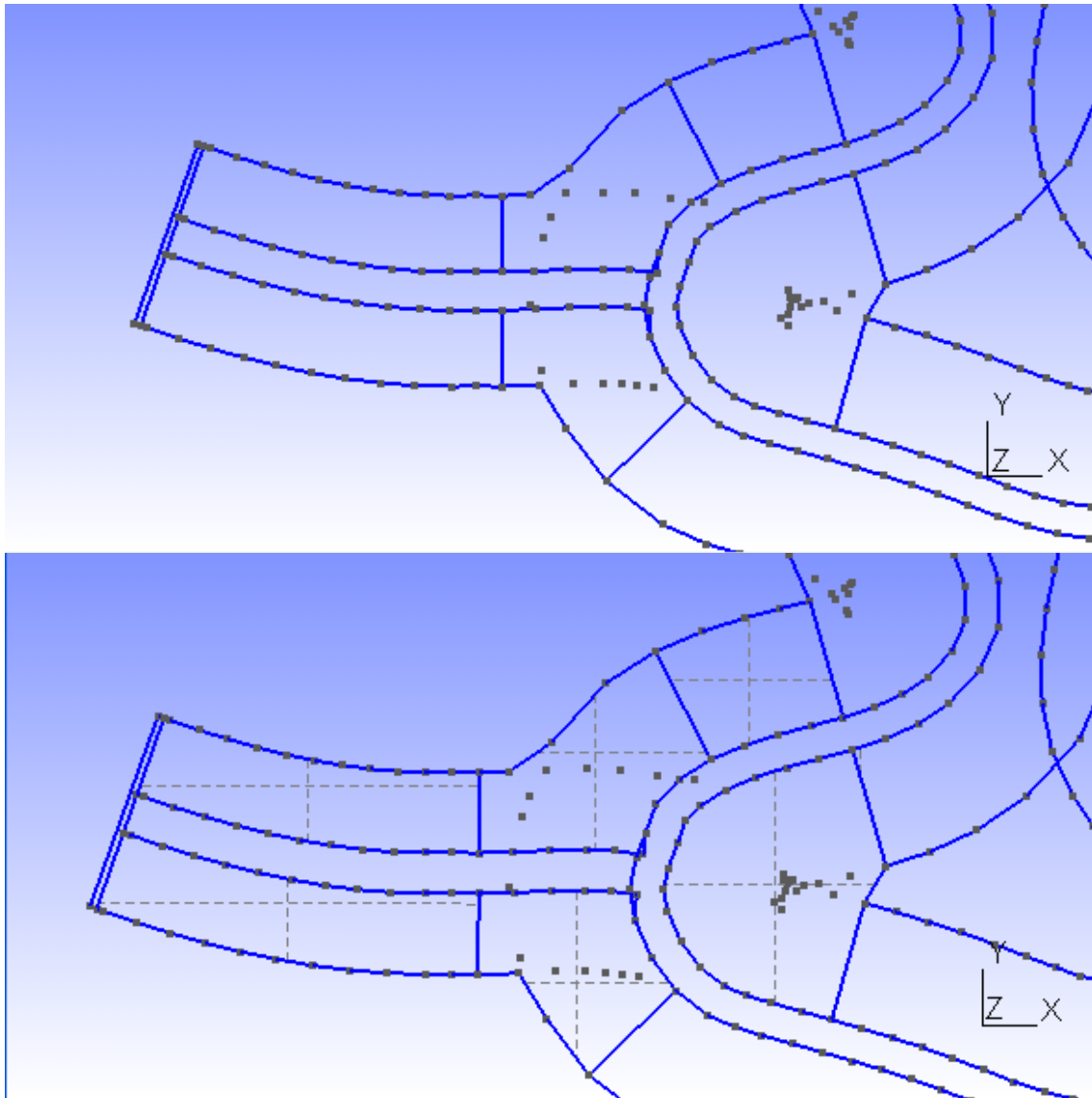


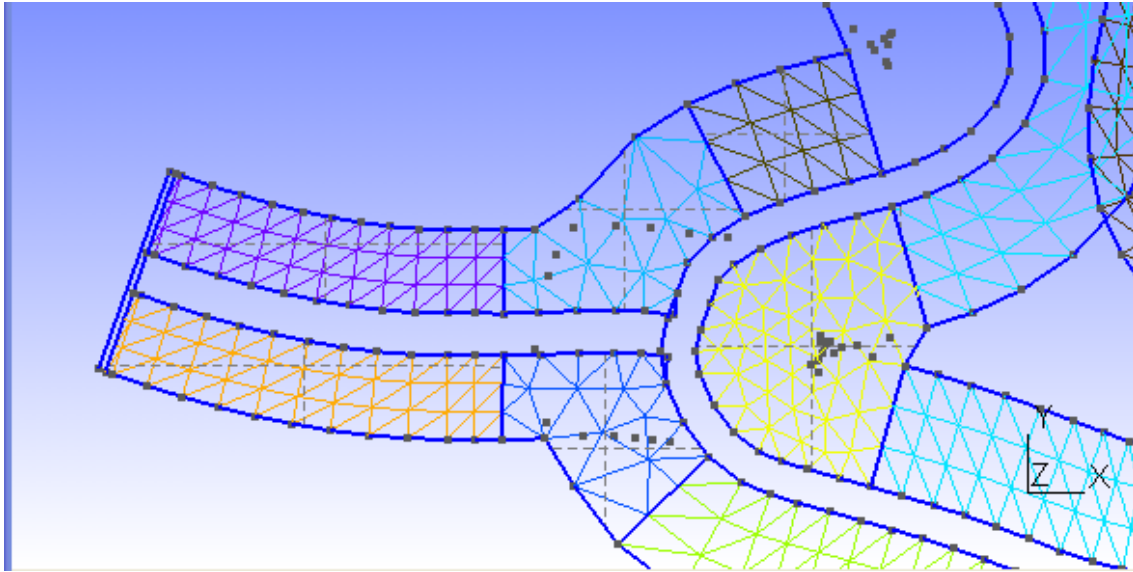
As you can see, the created surfaces have a regular pattern.

Now you can do the same steps for all the detours, or you can proceed to finish this one.

If you want to do complete all the steps for this detour, just proceed as explained in “the easy-way”: delete unwanted lines, create the needed ones and create the plane surfaces needed (only the ones with an irregular pattern).







2) Create Physical Surface(111)

Script join_geos already included a list of the driveable surfaces inside joined.geo. But we have deleted some of them and created a few. If you open joined.geo with a text editor, all the Plane Surface declarations created after the text line

```
//***** END OF JOINED .GEO FILES*****
```

must be included in the Physical Surface(111) list. So copy the Physical Surface declaration to the end of the file and add all the new surfaces to the list. To help in this task, you can run a script called “**listc**” that creates a file called “**listc.geo**”, with the list of plane surfaces created after the text line shown above.

```
octave-3.2.3.exe:11:f:\vmdata\tramos\tudons\tudons_father\s1_mesh
> listc
Leyendo el fichero salida\joined.geo
Cerrando el fichero
Escribiendo el fichero salida\listc.geo
Cerrando el fichero
octave-3.2.3.exe:12:f:\vmdata\tramos\tudons\tudons_father\s1_mesh
>
```

3) Create non-driveable zone and Physical Surface(222)

Non-driveable zone and Physical Surface must be created as usual: an external boundary with a hole boundary that is the outside limit of the driveable zone. When the surface is created the user has to add the Physical Surface(222) definition

4) Final check

Once you think joined.geo is ready, open it with gmsh, Mesh\2D, File\Save mesh and open

joined.mesh with gmsh. Check the mesh: check all the route looking for missing surfaces. If everything is ok go on with the process (trocea_malla).

VIII BACKGROUND IMAGES

VIII.1 Basics

The scripts can help you to include background images into your BTB project. If you want you can even use those images as the texture for your non-driveable terrain.

To work with satellite images you need SASPLANET:

http://sasgis.ru/programs/sasplanet/SASPlanet_100707.zip

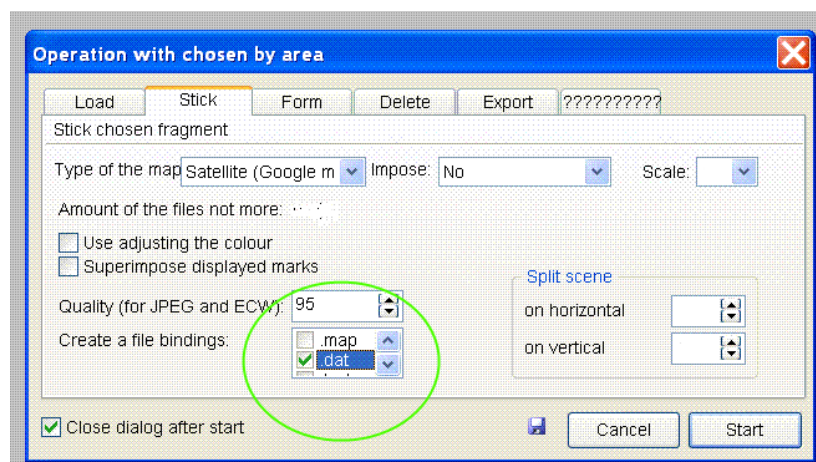
In step s4_terrain select the option "Create sasplanet.hlg". A file called s1_mesh\sasplanet.hlg will be created with the coordinates of a box that comprises your terrain.

Once your s4_terrain is completed you can open the file s1_mesh\sasplanet.hlg with **SASPlanet**:

Operations -> Select -> Load from file -> Load -> Start

When the process ends, save the images in a folder (c:\example_folder) with the desired splitting (5x3 grid, for example). **IMPORTANT: if you want to use the images as texture for the terrain you will have to split your terrain using exactly the same grid (step s10_split).** Use "**sat**" as the prefix for the images: they will be named sat_1-1.jpg, etc.

Operations -> Select -> Previous Selection -> Stick -> Start (YOU MUST select .dat file bindings)



The satellite images have been splitted and the info about the splitting coordinates is in the created .dat files.

In s10_split step you will have to say in what folder can be found the .dat files. In s10_split step the scripts (add_dat_to_geo) create a list of all the background images, with the format needed by BTB. It is called **s1_mesh\list_bi.txt**. If it exists, it is automatically inserted inside the Venue.xml when join_all is run. **If you don't want to use images in your project you should remove or rename this file before running join_all.**

Images should be named **sat_X-Y.dds** (please note that they are in .dds format) and should be located inside **My Project\XPacks\Common\Textures** (*My project* is your project's BTB folder).

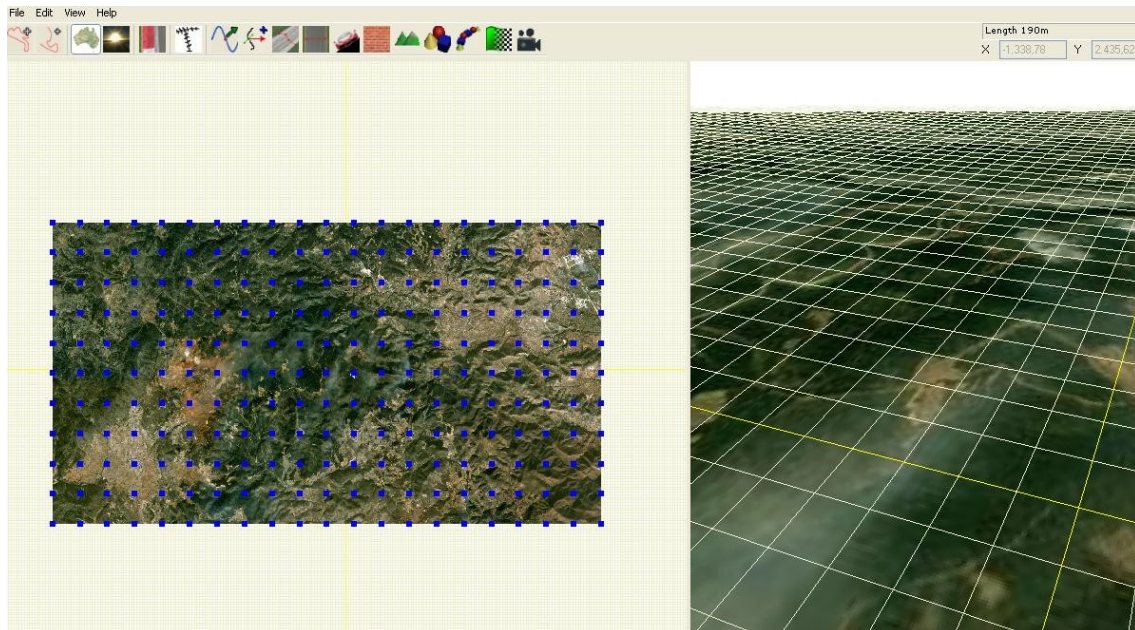
VIII.2 Blending with background images

In the creation of the BTB terrain, the scripts (procesar_elementstxt_mt) can automatically use the background images as texture for the non-driveable zone, if the terrain is splitted **using a grid of the same size as that used in SASPLANET (to create list_bi.txt).**

For example:

procesar_elementstxt_mt(10,5,0) splits the terrain using 10x5 grid, but does NOT map background images to terrain zones. If list_bi.txt exists and the images are located inside Common\Textures folder, the project will have background images, but they will not be linked to the terrain (as their texture).

procesar_elementstxt_mt(10,5,1) splits the terrain using 10x5 grid, using background images for blending in terrain zones. Before opening the Venue.xml it is mandatory 1) to create list_bi.txt (giving the right location for the .dat files) before Venue.xml is created and 2) copying the images to Common\Textures folder. Otherwise BTB will refuse to open your project.



Example list_bi.txt

```

<BackgroundImages count="50">
  <BackgroundImage>
    <Path>Common\Textures\sat_1-1.dds</Path>
    <Plane>
      <Position x="-11232.375390" y="-0.5" z="5636.028398" />
      <Scale x="2100.341106" y="1" z="-2311.287859" />
      <Rotation x="0" y="0" z="0" />
    </Plane>
  </BackgroundImage>
  <BackgroundImage>
    <Path>Common\Textures\sat_1-2.dds</Path>
    <Plane>
      <Position x="-11232.375390" y="-0.5" z="3324.740539" />
      <Scale x="2100.341106" y="1" z="-2311.287859" />
      <Rotation x="0" y="0" z="0" />
    </Plane>
  </BackgroundImage>
  etc.
</BackgroundImages>

```


VIII.3 Resumen

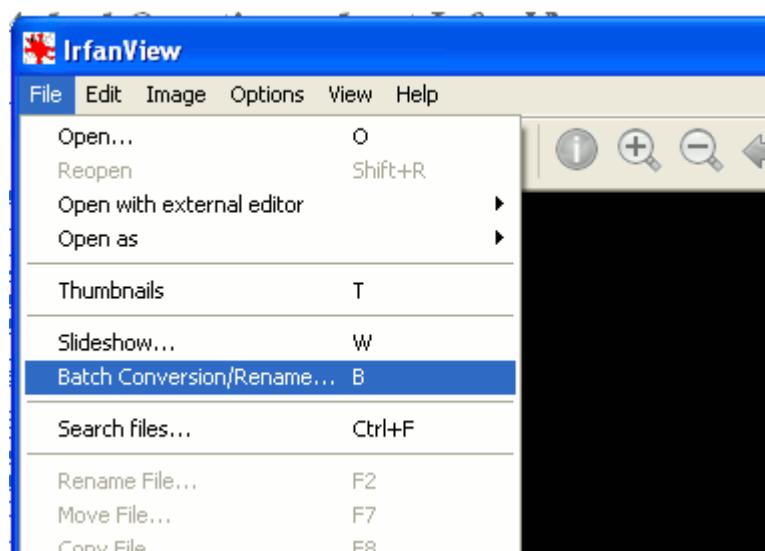
To get the images with SASPlanet you need a .hlg file with the info of your mesh boundaries.

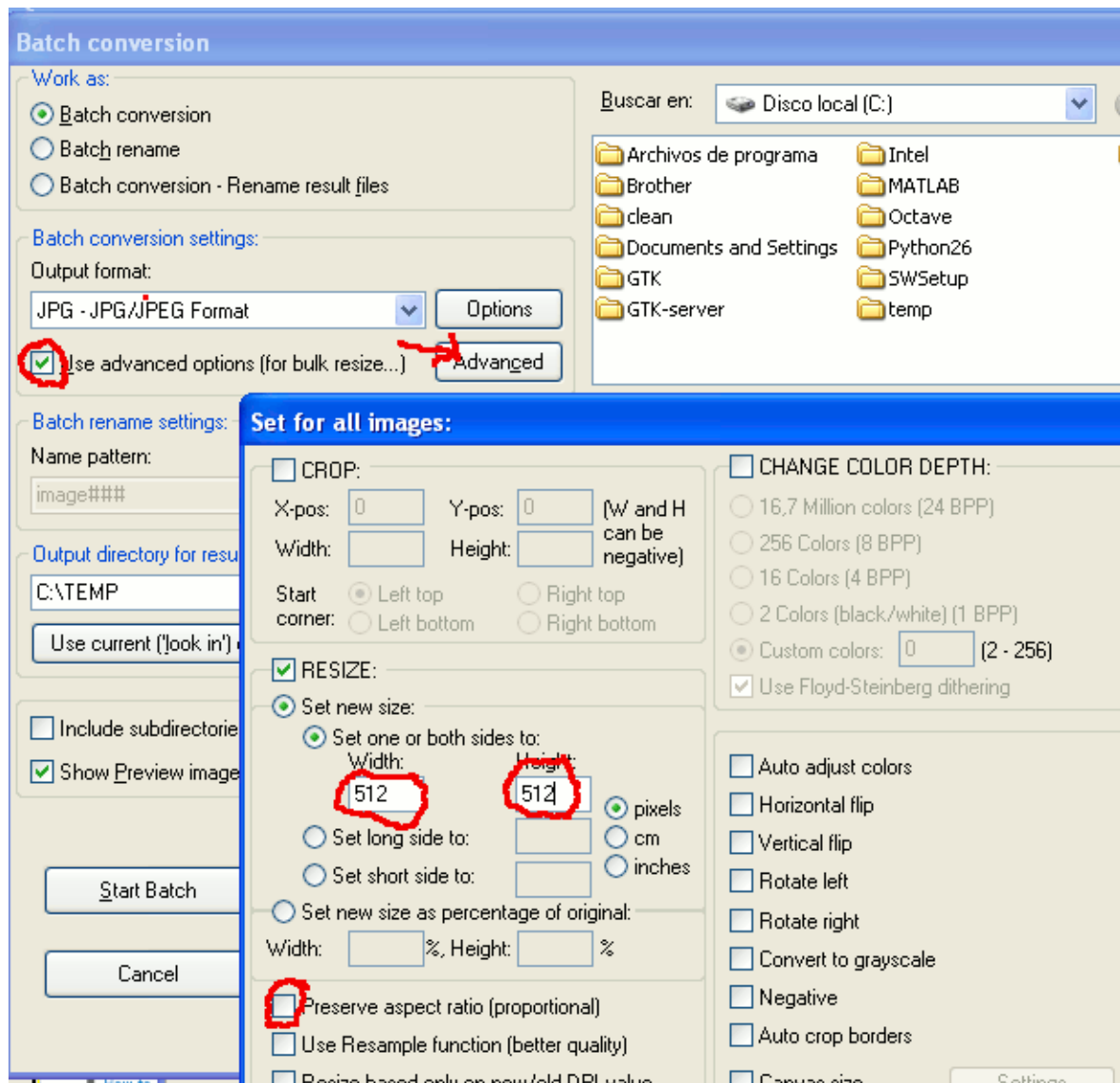
Selecting the right option on step s4_terrain (thus calling **create_hlg** script) makes that task: reads the mesh info (created from your .geo, saved as .msh and splitted with trocea_malla) and writes sasplanet.hlg

Then you open sasplanet.hlg with SASPlanet, following the steps explained in this chapter, and you get the .jpg files. For each .jpg image a .dat file has to be saved. This .dat files contain the terrestrial coordinates of the image limits. In step s10_split, a script (add_dat_to_geo) can read the .dat files and translates the terrestrial coordinates to BTB coordinates and creates a file ready to be inserted in the Venue.xml so the images are automatically placed in the background of the track. The output file is list_bi.txt. This file will be inserted in step s9_join(by script join_all) into the Venue.xml and you will get your images as background.

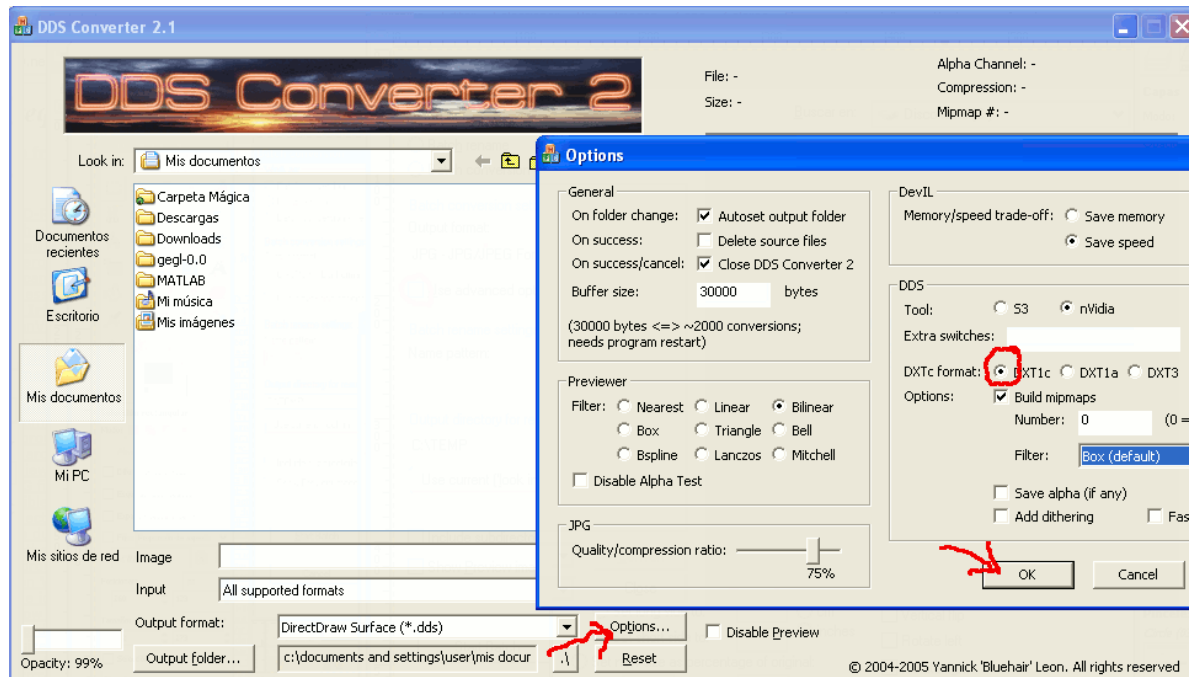
That above is a simple way to proceed. Now suppose you want to complicate things: you want a project with a terrain configured to make blending between a default texture and the background images (If that is the case, you must make sure you have created list_bi.txt by typing the right folder for the .dat files in s10_split step). Ok then, it is really easy: at the s10_split you just have to select option “Blend with background”(that calls procesar_elementstxt_mt with 3 parameters: dimensions of the matrix of images, and "1", to tell the script you want blending with background images, e.g procesar_elementstxt_mt(10,5,1)).

NOTE: I use Irfanview to batch resize .jpg with 2 power size, and DDSConvert to batch convert them to .dds. In Irfanview I select the “batch conversion” option and then I set the dimensions desired for the images (I unmark the “preserve aspect ratio” option).





In DDSConvert I select the DXT1c format (background images shouldn't have alpha channel).



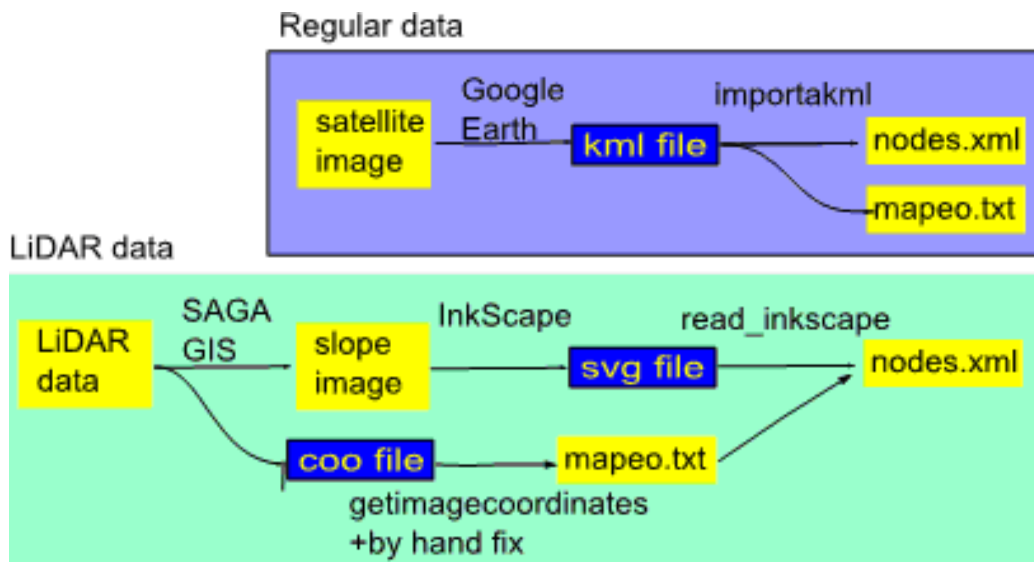
IX USING LiDAR DATA

Example of files used with InkScape (waste previous folder) and with the scripts:

http://www.mediafire.com/file/nu5hgpf9jbs4b24/waste_part1.7z

http://www.mediafire.com/file/gmy4n4xmzc5616s/waste_part2.7z

This type of elevation data has its own section on this document because they are treated in a special way. The difference with a normal project created with the scripts, is that as we have very good elevation resolution, we are **not** going to trust .kml routes created from satellite images. Instead of that **we will use elevation data to create an image** that shows us where the roads are and **we will create routes that match the roads on the image**. This way roads will be placed with high precision.



Step 1)

Use SAGA-GIS (<http://sourceforge.net/projects/saga-gis/>) to import .ASC files

- Import/Export - Grids\Import ESRI Arc/Info grid
- Terrain Analysis-Morphometry\Local Morphometry

If you have .las files, you can also use SAGA-GIS to import the data. For example, opening SAGA and following these steps:

- Import/Export-LAS\Import LAS Files
- Shapes\Point Cloud to Grid (1m grid)
- Grid Tools\Close gaps
- Terrain Analysis – Morphometry\Local Morphometry (slope)

And finally export the “**slope**” data using .jpg format:

- Import/Export Images\Export Image

Step 2)

Use FUSION (http://forsys.cfr.washington.edu/fusion/FUSION_Install.exe) to transform all the individual ASCII grids to DTM grid format

```
c:\FUSION\ASCII2DTM N4414A3.dtm m m 1 0 0 0 N4414A3.asc
```

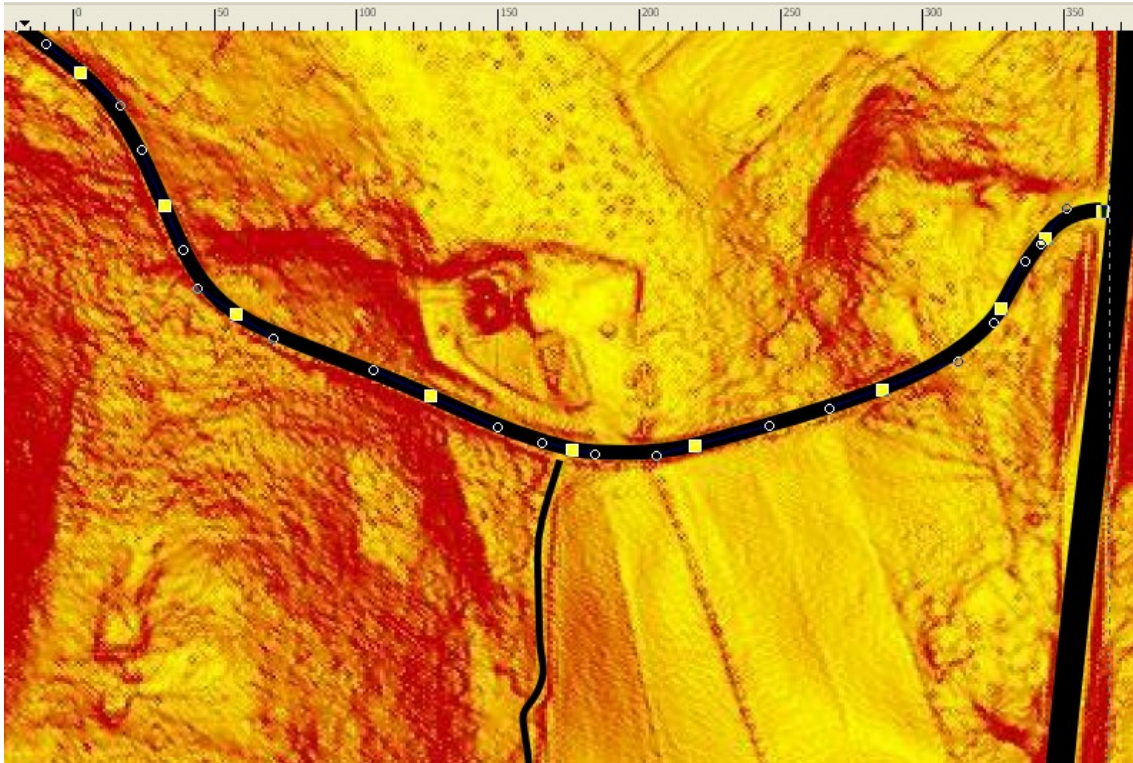
.dtm files must be copied in a folder called “lidar” placed in the same folder where we have the project folder or the father’s and sons’ folders in multitrack projects (e.g. if we have c:\project, then copy them to c:\lidar).

If we have a .las file I recommend filtering data to remove vegetation rebounds (example shown filters data with feet units to create 1m grid. If working with meters use 1 instead of 3.28084):

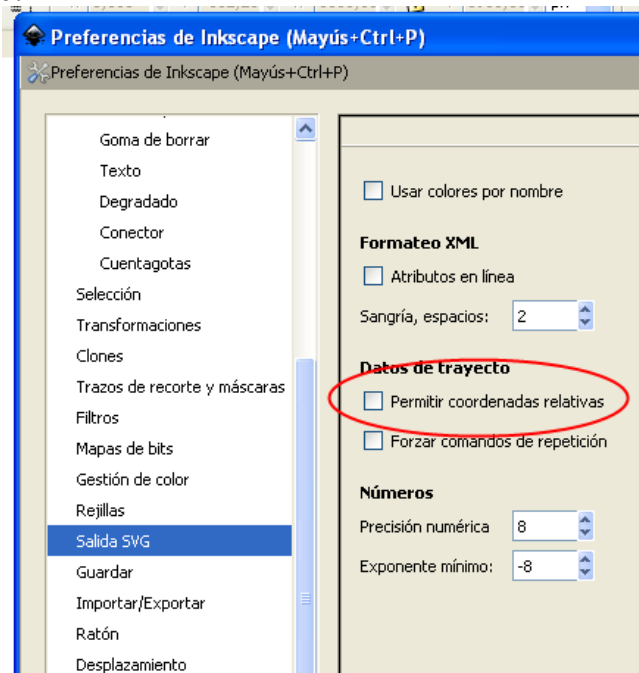
```
c:\FUSION\GroundFilter %1f.las 3.28084 %1.las  
c:\FUSION\GridSurfaceCreate %1f.dtm 3.28084 f f 1 0 0 0 %1f.las
```

Step 3)

Import the **slope** .jpg with InkScape (**don’t resize or move the image**) and create routes following the bright paths (low slope zones) that show where roads are. Create a path for each track in the project. Adjusting width in pixels to match the bright zones you can have a hint of the real width of the road (you may need that value later when using btb06).



NOTE: In the **general options** of Inkscape, **SVG output** section, select that relative coordinates are **not** allowed. This way the bezier curves should contain only *M* and *C* letters, and not *m* or *c*.



A close-up of the software toolbar. The buttons are labeled 'Objeto', 'Trayecto', 'Texto', and 'Filtros'. The 'Trayecto' button, which features a blue square icon with a white path, is circled in red. Below the toolbar, a portion of a coordinate system is visible, showing values -500, -400, and -300.

$$\begin{array}{l} \min x \ y \ z \ X1 \ Y1 \ Z1 \\ \max x \ y \ z \ X2 \ Y2 \ Z2 \end{array}$$

-68-

files can be renamed as nodes.xml, copied to the Venue folder of the scripts and then processed with btb06. We repeat the process for all the routes, for example if it has 5m width:

```
> cd Venue
> btb06(5,1)
```

And the project can be processed as usual with the GUI, from s3_road step.

But **create_sons** can do part of that step for us, as explained below. Before using the scripts mapeo.txt file has to be placed in the root folder of the project (or root folder of the father). Then we can go on with the use of the scripts the same way as if we had used importakml.

Step 6)

I download the scripts using subversion. I copy mapeo.txt in the root folder of the father.

Now **create_sons** can create the sons automatically from the .xml files located inside a folder (e.g. *create_sons('c:\temp\pennsylvania')*). It also copies the .xml files as Venue\nodes.xml for each son.

Nevertheless we can also do that task by hand:

- Use create_sons(N) in the root folder of the father to create N sons
- Copy .xml files as nodes.xml inside Venue folders of father and sons

Step 7)

Use the scripts normally. FUSION must be installed in C:\FUSION. FUSION will use the .dtm files to give elevation to the points when needed.

For **creatrack1** it is recommended to use 1 as parameter: **creatrack1(1)**. This makes road elevation be calculated for points in the center line of the road. Otherwise road elevation profile is calculated taken into account points on the boundaries of the road.

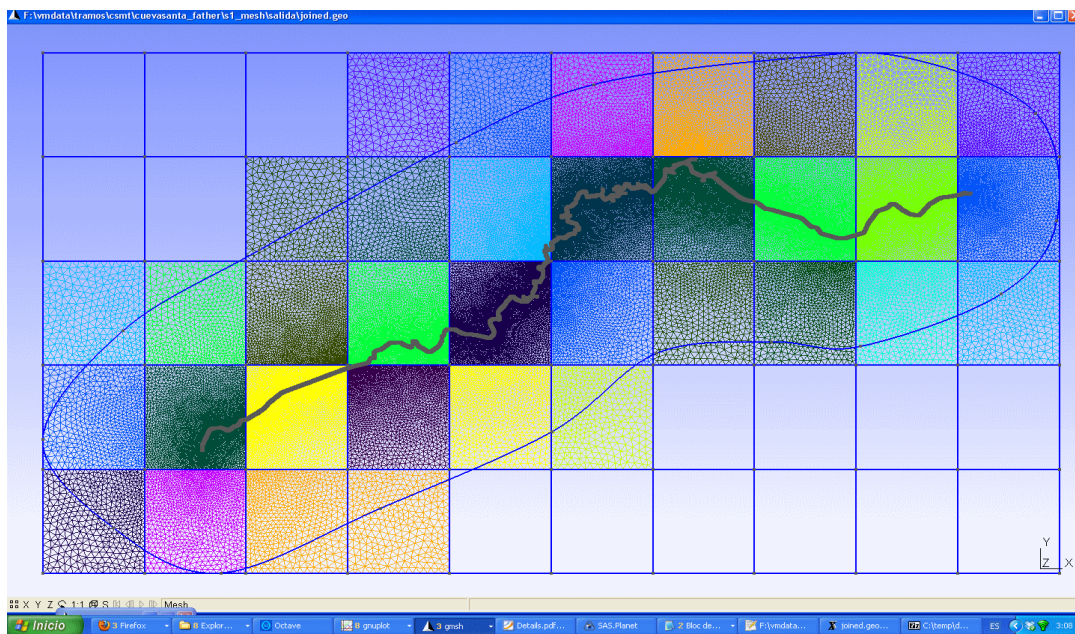
NOTE: LiDAR data can be downloaded for a few USA states from:

http://lidar.cr.usgs.gov/LIDAR_Viewer/viewer.php

X ADVANCED USES OF THE SCRIPTS

X.1 Terrain that matches background images

You want to have terrain areas that fit exactly your background images, as I did with my Cueva Santa track. Ok, if that is the case, then first you need to insert the limits of your images into the joined.geo, the .geo you are working with. It is really simple: you run `add_dat_to_geo` inside `s1_mesh` folder and you select the option "add the grid to joined.geo". Then when you open joined.geo with gmsh and you will see the grid with the limits of the images you downloaded with SASPlanet. You have to do a little work merging those new lines with your existing terrain, creating smaller surfaces for the non-driveable zone. It is not difficult to do this step if you have a little practice using gmsh. In the image it can be seen the original mesh limits (the biggest spline curve), and the new meshes fitting the limits of the downloaded images.



The Plane Surfaces that do not touch the driveable surface have no problems. But for those that overlap with that terrain you can follow these steps:

- 1) Make sure you are using the TOP view clicking on the Z (low-left corner)
- 2) Create a new point near of the terrain so the point belongs to the a line that touches the driveable terrain
- 3) Delete that line
- 4) Create a new line, the substitute of the one you just destroyed, but this one ending at the

point you just created, therefore not touching the driveable terrain

5) Create a line from your new point to the nearest point, in the driveable terrain boundary.

Try to keep this line aligned with the line you just created

6) Repeat the steps above for all the lines with problems

6) Create the plane surfaces and add their codes to the physical surface 222 list

Once you have your new mesh, you go on with the process as usual.

XI LIST OF COMMANDS

| Script | Run it in directory |
|--|---------------------|
| accept_mesh.m accepts anchors_carretera.msh as the definite mesh, skipping further processing with MeshLab. To be run instead of simplificar+MeshLab+juntar_mallas | s4_terrain |
| add_dat_to_geo.m Reads the .dat files created by SASPlanet while splitting a satellite image and creates a grid that can be saved as grid.geo or appended to salida\joined.geo. This script also created a list of background images (s1_mesh\list_bi.txt) ready to be included in the Venue.xml. NOTE: this script internally calls addgrid, so addgrid.hlg is overwritten. | s1_mesh |
| addgrid.m Creates a grid with .geo format. Two possibilities: addgrid(numx,numz) You want to view the available elevation data using a numx X numz grid addgrid(xmin,xmax,zmin,zmax,step) Creates a grid with the specified limits and line separation. If another parameter is added to the command, no matter its value, the list of points and lines created will be explicit (instead of using a “for loop”). addgrid creates a file called addgrid.hlg ready to be opened with SASPlanet to get the satellite images for that area. | s1_mesh |
| add_subject.m Creates a list of SObjects to be inserted by hand in the Venue.xml. add_subject(num_points) Parameter is the maximum number of points used by one SObject. Longer SObjects will be splitted. | s7_walls_b |
| addt.m Opens joined.geo and replaces the last occurrence of a Plane Surface followed by a Spline with the code to define that surface as Transinite | s1_mesh |

| | |
|--|-----------------------|
| btb_a_coor.m Returns the terrestrial coordinates of a BTB point <pre>> [mapeo]=textread('mapeo.txt','%f'); > x=2380.47; > z=-2350.67; > [longitud altura latitud]=BTB_a_coor(x,0,z,mapeo)</pre> | base directory |
| btb06.m Creates the points in both borders of the road, where the road and the terrain will be linked (they are called anchors). Parameter is the separation between anchors on right and left side. It will affect the mesh created by <code>mallado_regular</code> | venue |
| coor_a_btb.m Returns the BTB coordinates of a point given the terrestrial coordinates <pre>> [mapeo]=textread('mapeo.txt','%f'); > longit= -73.67; > latit= 41.47; > [x1 y1 z1]=coor_a_BTb(longit,latit,elevation,mapeo)</pre> | base directory |
| corregir.m For a given road, compares the elevation profile assigned using <code>dar_altura</code> and that obtained from elevation data (from <code>lamalla.mat</code>), and changes the terrain elevation data (<code>lamalla.mat</code>) to fit the elevation profile set with <code>dar_altura</code> . <code>corregir</code> also accepts a <code>kml</code> file as a parameter and uses its coordinates and altitude to change <code>lamalla.mat</code> . This could be useful if we have a <code>kml</code> with altitudes we trust, but dangerous as those altitudes could have an offset respect to the elevation data available. <pre>corregir('file.kml')</pre> | s3_road |
| creartrack1.m Gets elevation values for a road from its coordinates and elevation data (<code>lamalla.mat</code>) | s3_road |
| create_hlg Creates file <code>s1_mesh\sasplanet.hlg</code> (open it with <code>SASPlanet</code>) with the boundary coordinates (box) of <code>anchors_carretera.msh</code> (run <code>trocea_malla</code> before using <code>create_hlg</code>) | s1_mesh |

| | |
|---|--|
| <p>create_sons Creates the folder's structure for several sons in a multi-track project.</p> <p>create_sons(number_of_sons) Creates son01, son02, etc. folders in the same folder where the father is located</p> <p>create_sons('c:\temp\kmls',keep_names) Creates one son for each kml located inside the directory used as first parameter. If keep_names is 0, folders will be named son01, son02, etc. If keep_names is 1, folders will keep the name of their respective kmls</p> | <p>father's root directory</p> |
| <p>cut_lamalla.m Reduces the size of lamalla.mat. Useful if data comes for a too big zone.</p> <p>cut_lamalla([xmin xmax],[zmin zmax])</p> | <p>s2_elevation s2_elevation_b</p> |
| <p>dar_altura.m Softens the output from creartrack1 and gives the nodes of the track their elevation and slope to fit that curve.</p> <p>dar_altura(smooth_factor,pos_slope,neg_slope,step,interactive)</p> <ul style="list-style-type: none"> - smooth is a smoothing factor, the bigger the smoother - pos_slope and neg_slope are the maximum and minimum slopes allowed (1 means 45 degrees) - the final elevation profile is constructed using one point each "step" meters. Use a small value to preserve the profile's details, and a big value to smooth them. 25m is used if omitted - If interactive==0, the script doesn't give the user the option to edit the profile by hand and exists | <p>s3_road</p> |
| <p>fix_project.m This script reads tracks' point coordinates contained in joined.geo and creates new files porcentajes.mat and anchors.mat for all the tracks in the project. This way may be a project where tracks and terrain are not correctly linked any more can be fixed. After running this script, all the steps from juntar_mallas to the end must be redone. This script won't work correctly if sons have been added or removed since joined.geo was created. Make a backup before using this script.</p> | <p>s1_mesh\salida</p> |

| | |
|---|--|
| importakml.m Reads a kml file and from it creates a mapping between terrestrial and BTB coordinates. importakml(kml_file) All the original points of the kml will be converted to nodes of the road importakml(kml_file,'decimate',factor) Keeps 1 from every “factor” points of the kml as nodes of the road. For example if the kml has 100 points and factor==2, the road will have 50 nodes. importakml_old(kml_file,tolerance) Uses the old “approach”. An <i>ideal</i> smooth road with a huge amount of nodes that follows the coordinates of the kml file (using akima splines). Finally some nodes are removed. A node is removed if removing it doesn't deviate the road more than “tolerance” meters from the “ideal path” | s0_import |
| join_all.m Final step of the process. Joins all the tracks, terrain, pacenotes and walls, creating a file called Venue.xml. To open this file good luck and WP.zip Xpack are needed. | s9_join |
| join_geos.m Joins the anchors_carretera.geo files created with mallado_regular for all the projects, creating file joined.geo inside s1_mesh\salida folder. This file should be edited with gms. | s1_mesh |
| juntar_mallas.m Reads i.ply, c.ply and n.ply from s4_terrain\salida and joins them in one single mesh (files anchors_contaltura.txt and elements.txt) | s4_terrain |
| leehgt.m Creates lamalla.mat from a .hgt file (1 degree x 1 degree) leehgt(fichero,latitud,longitud) Data extension is from latitud to latitud+1 and from longitud to longitud+1 | s2_elevation s2_elevation_b |
| leehgt2.m The same as leehgt, but joins 2 adjacent .hgt files leehgt2(file1,latit1,longit1,file2,latit2,longit2) if latit1==latit2, longit1 should be <longit2 if longit1==longit2, latit1 should be <latit2 | s2_elevation s2_elevation_b |
| leer_gridfloat.m Creates lamalla.mat from gridfloat file. First parameter is the .hdt and second one is .flt | s2_elevation s2_elevation_b |
| leentif.m Creates lamalla.mat from a geotiff file | s2_elevation s2_elevation_b |

| | |
|--|--|
| <p>listc.m Reads salida\joined.geo and creates a file called listc.geo with the id numbers of all the Plane Surfaces created inside joined.geo after its creation (last line of joined.geo after its creation is the reference used by listc)</p> | <p>s1_mesh</p> |
| <p>make_grid.m Creates several files containing a regular grid of points with terrestrial coordinates. Those files should be “raised” with BTBLofly or a similar application and save with a different name: grid001.kml should be saved in the same folder as grid001_relleno.kml make_grid(xmin,xmax,zmin,zmax,step,file_size) Parameters are x and z minimum and maximum values, and distance between points of the grid. Maximum file_size depends upon the application to be used. 5000 is recommended for BTBLofly. Another possibility for make_grid is creating a kml route and asking make_grid to create a grid that covers all that route: make_grid('limits.kml',step)</p> | <p>s2_elevation s2_elevation_b</p> |
| <p>mallado_regular.m Creates a terrain mesh on both sides of the road. Position of road borders (anchors) is taken from btb06 output. Besides the road a terrain of a specified width will be created, splitted in the transversal direction into the desired number of panels. Terrain width (meters) is the first parameter and the number of panels is the second one. If you want to try a regular pattern (tranfinite) for all the driveable zone, use 1 as 3rd parameter. Otherwise use just 2 parameters</p> | <p>s1_mesh</p> |
| <p>muro_pegado.m Creates walls on both sides of the road (from start to end). List of walls can be found in salida folder and should be inserted by hand inside the Venue.xml file (updating the total walls count, if needed) muro_pegado(tam_wall,offset) Parameters are the limit of points per wall and the displacement in meters in the outside direction from the road border (the width specified as btb06 parameter is used to compute border position)</p> | <p>s7_walls_b</p> |
| <p>msh2btb Creates a BTB terrain from file s10_split\salida\anchors_carretera.msh. The created terrain will not be blended with background images and it won't be connected to the roads. This command assumes you use it instead of procesar_elementstxt_mt ply2btb(cells_x,cells_z) Will split the terrain using a cells_x X cells_z grid</p> | <p>s10_split</p> |

| | |
|---|--|
| pacenotes.m Gets the track shape from a driveline.ini file. Output from this script will be used by pacenotes_2 | pacenotes |
| pacenotes_a.m Gets the track shape from anchors created by btb06. Output from this script will be used by pacenotes2_a | pacenotes |
| pacenotes2.m Creates a new pacenotes.ini file using the old one and the output from pacenotes.m pacenotes2(sensibility,distance) Parameters are the sensibility for curve detection and the distance you want to move the pacenotes to the start of the road. 10 means 50m. | pacenotes |
| pacenotes2_a.m Creates a list of pacenotes in BTB format ready to be inserted inside the Venue.xml. Join.all looks for this pacenotes and if they exist, includes them inside Venue.xml. Parameters are the same as pacenotes2 | pacenotes |
| partir_track.m Splits a track into several segments. Reads split points from pos_nodes.txt | s10_plit |
| plot_lamalla.m Plots the contents of salida\lamalla.mat as a surface. | s2_elevation s2_elevation_b |
| ply2btb Creates a BTB terrain from file s10_split\salida\n.ply . The created terrain will not be blended with background images and it won't be connected to the roads. This command assumes you use it instead of procesar_elementstxt_mt ply2btb(cells_x,cells_z) Will split the terrain using a cells_x X cells_z grid | s10_split |
| poner_muro.m Creates walls in the boundary between driveable and non-driveable zones. Walls are automatically included inside Venue.xml by join_all | s7_walls |
| procesar_elementstxt_mt.m Creates the terrain in BTB format from the mesh created by juntar_mallas and the output from partir_track. By default terrain is splitted using a 10x10 grid, but user can choose another grid size. procesar_elementstxt_mt(cells_x,cells_z,do_mapping) Will split the terrain using a cells_x X cells_z grid, and If do_mapping is 1, terrain will be created with background images blending (see add_dat_to_geo). | s10_split |

| | |
|--|--|
| procesar_nodostxt.m Nodes of anchors_carretera.msh mesh receive a elevation value taken from lamalla.mat, if possible, or lamalla2.mat | s4_terrain |
| process_sons.m This script processes all the sons in a multitrack project. It should be first edited to set the desired values for the parameters of the scripts called. | base directory of father |
| raise_kml.m Calls a Google Earth API to get elevation values for the gridXXX.kml files inside s2_elevation\salida folder. Output files will be named gridXXX_relleno.kml and will be ready to be processed with read_grid This script needs Google Earth and Python27 installed in the system, Read instructions for installation inside documentation folder. | s2_elevation s2_elevation_b |
| read_grid.m Reads the gridXXX_relleno files and created lamalla.mat, with all the elevation info collected | s2_elevation s2_elevation_b |
| readkml.m Translates a route from a kml file to a curve in gmsh format and BTB coordinates. Output file is written in salida folder, with the same name as input, but .geo extension. readkml('file.kml',curve) Second parameter can be "t", for adding straight lines, "s" for adding a spline, or "st" for adding both. Not using a second parameter means adding no curve (just points). | s1_mesh |
| readkml_bat.m Calles readkml for all the .kml files found in the specified folder. readkml bat('d:\folder',curve) | s1_mesh |
| simplificar.m Splits anchors_carretera.msh in three parts that should be processed with MeshLab: intocables.ply, conducibles.ply and noconducibles.ply Also creates a folder ncSplitted with a separate .ply file for each surface of the non-driveable zone, so it is possible to simplify them individually. | s1_mesh |
| split_agr.m Splits a file with extension .ASC into several smaller files with the same format but extension .AGR. Ejemplo: split_agr('MDT05-0667-H30',5) splits MDT05-0667-H30.ASC file into 5x5=25 files with extension .AGR | agr |
| split_track.m Selects the points for splitting a track into several segments. Writes those points in file pos_nodes.txt, allowing the user to change them before running partir_track | s10_split |

| | |
|---|--|
| <p>start.m</p> <p>Calls importakml to process s0_import\road.kml Calls make_grid with s2_elevation\limits.kml as parameter Calls make_grid with s2_elevation_b\limits_b.kml as parameter Default steps for make_grid are 25 and 75m, respectively. By default each gridXXX.kml is limited to 5000 points</p> <p>start start(step,step_b,file_size)</p> <p>Parameters are the steps in meters used by make_grid and the number of points per kml.</p> | |
| <p>terrain_noise.m</p> <p>Adds a random value to the elevation of the nodes of the terrain. Random value will be in the range specified. Use this script just before join_all</p> <p>terrain_noise([ymin ymax])</p> | s4_terrain |
| <p>trocea_malla.m</p> <p>Splits anchors_carretera.msh into 2 parts: list of mesh nodes (nodos.txt) and triangles (elements.txt)</p> | s1_mesh |
| <p>vercontorno.m</p> <p>Shows a contour plot using the terrain elevation data (lamalla.mat) and the road position (output from btb06)</p> | s2_elevation s2_elevation_b |

XII ADDITIONAL NOTES

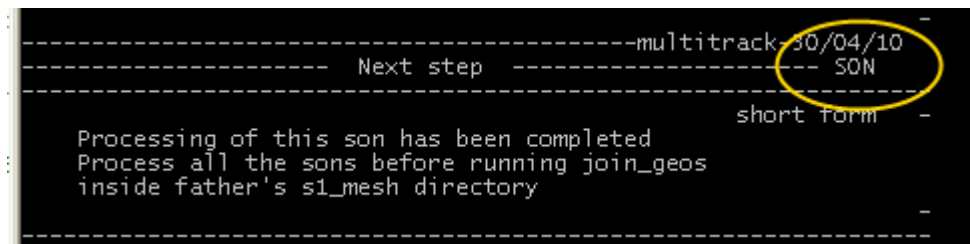
XII.1 GUI

Sometimes the interface refuses to perform an action for no reason. If that happens, close the GUI, open it again and try again. If that doesn't work you can close the GUI and type the commands on the text-mode octave window. The error messages on screen will help to diagnose the problem.

If you get an error message related to iconv.dll may be copying that file from "c:\GTK\bin" to "c:\GTK-server" can solve it.

XII.2 Gmsh and multitrack

- 1) If you don't see the **key word SON or FATHER on the "Next step message"**, don't go on with the process until you correct the problem



```

-----multitrack s0/04/10
----- Next step ----- SON
-----
short form
Processing of this son has been completed
Process all the sons before running join_geos
inside father's s1_mesh directory
-----

```

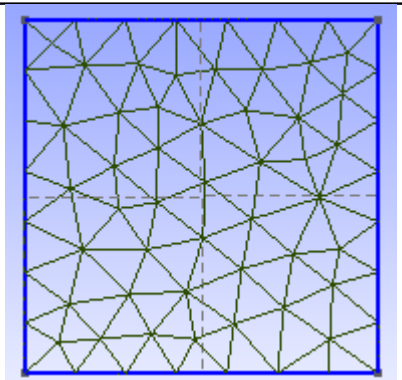
- 2) **Crossings** can be processed easier if they are treated as **2 separate branches**. So when creating the kml files for them just create 2 kmls instead of 1.
- 3) There is a script called **process_sons** that can be used to process all the sons if there are a lot of them and they share parameters like road width or mesh width and number of panels. File scripts\process_sons.m can be easily edited to fit your needs. To use this script run first "cd C:\project_father". Please note that this script uses the first .kml file it finds inside each s0_import directory (so save there only one .kml file).
- 4) When you create or delete an object with gmsh, this program annotates the change immediately inside the .geo file. If you want to undo steps, you can close gmsh, open the .geo file with a text editor and remove the unwanted operations.
- 5) With gmsh you can't delete a point that belongs to a line. You can't delete a line that

belongs to a surface. If you want to remove a line that belongs to a surface, first remove the surface.

- 6) Can I have a “island” of non-driveable terrain, completely surrounded by driveable terrain? I believe poner_muro will fail if that situation is given. You can create that surface as driveable and later with BTB split the terrain and change the properties. Nevertheless in that case no walls would be automatically created for that “island”.
- 7) Don't run **listc** after creating the Plane Surface for the non-driveable zone. If you do that and use the file, remember to remove that surface from the list or you will get that terrain zone duplicated inside your project.
- 8) I have processed joined.geo and now I want to add another track to the project, can I do that without redoing all that work done with gmsh? I think it is possible. Start by doing a backup. Then remove the line with text “**END OF JOINED .GEO FILES**” from joined.geo and insert at the final part of the file the anchors_carretera.geo created for the new son. Try to open it with gmsh. If it doesn't work, ask me.
- 9) What is the mission of script addt? Why do I create a spline, if I don't want a spline? Let's make clear that point. How do we create a surface with a regular pattern?

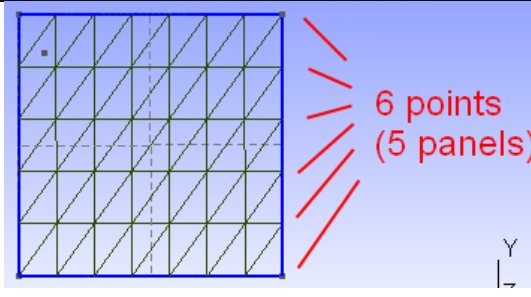
Ejemplo:

```
Point(1) = {-1, 1, 0}
Point(2) = {-1, -1, 0}
Point(3) = {1, -1, 0}
Point(4) = {1, 1, 0}
Line(1) = {1, 4}
Line(2) = {4, 3}
Line(3) = {3, 2}
Line(4) = {1, 2}
Line Loop(5) = {2, 3, -4, 1}
Plane Surface(6) = {5}
```



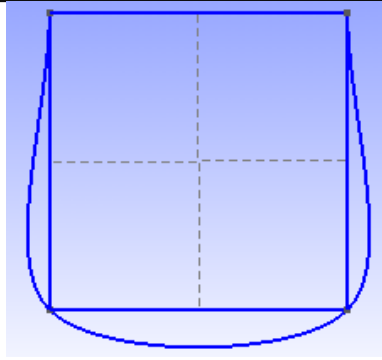
If we want to create a horizontal regular pattern, we must use Tranfinite lines (setting the number of points on them) on left and right lines and also declare the surface as Transfinite.

```
Point(1) = {-1, 1, 0};
Point(2) = {-1, -1, 0};
Point(3) = {1, -1, 0};
Point(4) = {1, 1, 0};
Line(1) = {1, 4};
Line(2) = {4, 3};
Line(3) = {3, 2};
Line(4) = {1, 2};
```



| | |
|--|--|
| <pre> Line Loop(5) = {2, 3, -4, 1}; Transfinite Line(4)= 6 Using Progression 1; Transfinite Line(2)= 6 Using Progression 1; Plane Surface(6) = {5}; Transfinite Surface(6)={1,2,3,4}; </pre> | |
|--|--|

Script “addt” (add transfinite) looks for the last occurrence of “Plane Surface” on joined.geo and if after the Plane Surface there is a spline declaration, it **changes the Spline for a Transfinite Surface** declaration using the same number as the plane surface and the same points as the spline list. For example, if the contents of joined.geo is:

| | |
|---|---|
| <pre> Point(1) = {-1, 1, 0}; Point(2) = {-1, -1, 0}; Point(3) = {1, -1, 0}; Point(4) = {1, 1, 0}; Line(1) = {1, 4}; Line(2) = {4, 3}; Line(3) = {3, 2}; Line(4) = {1, 2}; Line Loop(5) = {2, 3, -4, 1}; Transfinite Line(4)= 6 Using Progression 1; Transfinite Line(2)= 6 Using Progression 1; Plane Surface(6) = {5}; Spline(7)={1,2,3,4}; </pre> |  |
|---|---|

and we run “addt”, joined.geo would be changed to have the same contents and that shown on the previous code (that with the Transfinite Surface declaration).

That is the reason why the order of execution should always be: create the surface, create the spline and run “addt”. If you forget to run “addt” you can always open joined.geo later with a text editor, search for “Spline” and change it by hand.

If we don’t want to use “addt”, we can create the Plane Surface with gmsh, and then without closing gmsh open joined.geo with a text editor, add the Transfinite Surface declaration by hand, writing the numbers of the corner points of the plane surface that we want to give a regular pattern, and save the file. Next time we mesh we would see the result.

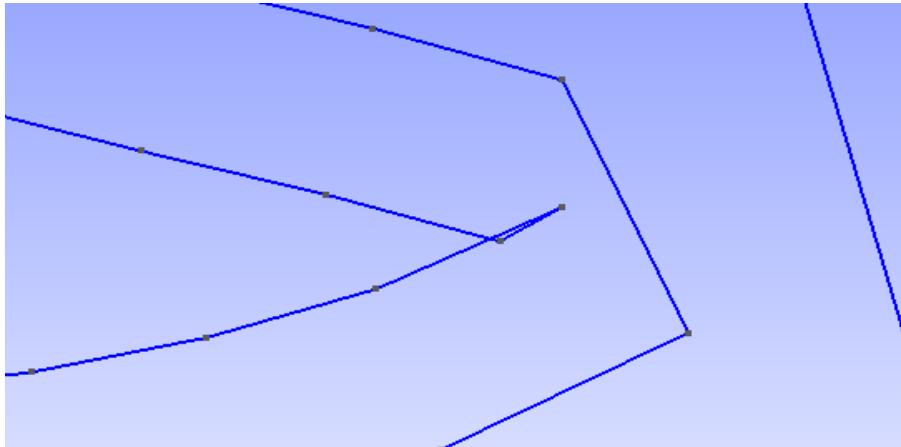
XII.3 Memory exhausted

Sometimes the terrain has too many faces and join_all, the last step, fails to create the Venue.xml. If that is the case, a little trick can be used to solve the problem:

1. Rename s10_split/salida/lis.txt as real_lis.txt
2. Create a file s10_split/salida/lis.txt with the text "LITTLE_TRICK" as its content.
3. Run join_all
4. Edit Venue.xml with a text editor, search for the text "LITTLE_TRICK", and replace it with the contents of real_lis.txt.

XII.4 Gmsh crashes while meshing 2D?

Maybe you are having problems with gmsh. You get an error message while trying to mesh-2D and gmsh crashes. If that is the case maybe the problem is that you have crossing lines (see picture below) in the road boundaries. Use the translate tool (see XII.6) or change the points coordinates other way to avoid the crossing.



XII.5 Can I add a kml route to gmsh?

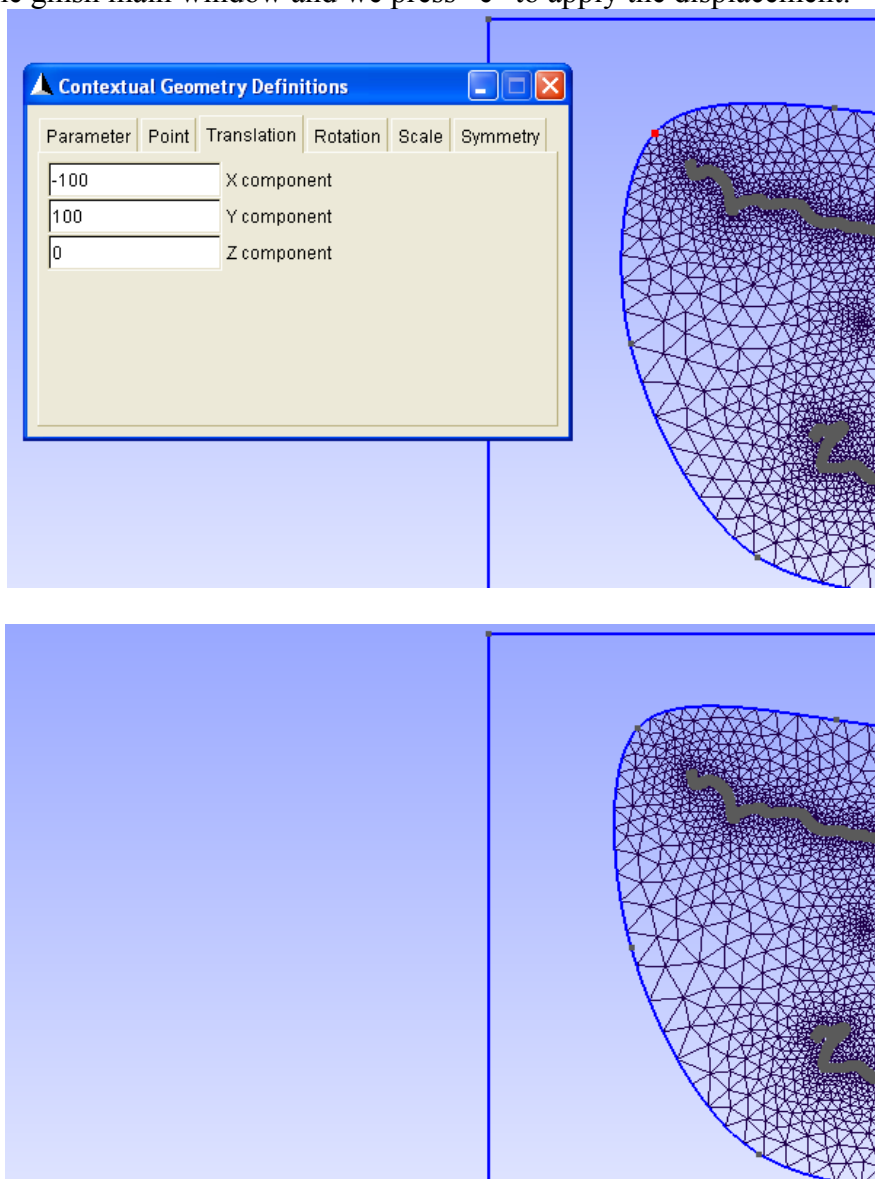
Yes, if you want to add set of points/lines to your .geo using as input a .kml file you can have a look at **readkml** script. You can add the output of this script to your .geo (anchors_carretera.geo or joined.geo) and you can use it to define your plane surfaces.

XII.6 Translate tool in gmsh

Sometimes when creating the meshes with gmsh we need to move a point. The tool we need is called “Translate”. Inside of gmsh we select Geometry->Elementary Entities->Translate->Point.

The mesh we create only has X,Y coordinates. Z coordinate must always be 0 so when we translate a point **displacement in Z direction must always be 0**.

We select the point to translate, we type the displacement in X and Y coordinates and then we select the gmsh main window and we press “e” to apply the displacement.



The changes are included in the .geo so they will be applied every time we open the .geo file with gmsh.

Example of code added to the .geo by the translate tool:

```
Translate {-100, 100, 0} {  
  Point{5644};  
}
```

XIII LINKS

XIII.1 Videotutorials

Install of the Grahical Interface

<http://www.mediafire.com/?myip7mte8rjgpgm>

(extract files and open gui.htm with a web browser. Flash player required.)

(Wink Source: <http://www.mediafire.com/file/w0zmi53e8cu9nm2/gui.wnk>)

2 Tracks using AGR elevation data

Videotutorial 1/3: http://www.mediafire.com/file/afdod089q47dgvc/agr_example.7z

(wink source: http://www.mediafire.com/file/da7rj6xvvdfa8p0/agr_source.7z)

Videotutorial. 2/3: <http://www.mediafire.com/?8d0s63wqaqu24xh>

(wink source: http://www.mediafire.com/file/kb8t9gqd93kghk9/agr_source_b.7z)

Videotutorial. 3/3: http://www.mediafire.com/file/7rsbpi46lf207ya/agr_example_c.7z

(wink source: http://www.mediafire.com/file/y263vv5g4yuiwn4/agr_source_c.7z)

Parlesportes' videotutorials.

They are great, even if you don't speak french language.

<http://www.mediafire.com/?tspub5ub5ncqe>

liquido's videotutorial (gmsh step for a one-track project)

Spanish audio:

<http://www.mediafire.com/?fcgfy67999cdal3>

(also available in Vimeo: <http://vimeo.com/37193937>)

English subtitles:

<http://www.mediafire.com/?7k5qd59cqbbiyi4e>

XIII.2 Forums

- 1) [Méthode zaxxon in Rallyesim forum](http://forum.rallyesim.fr/viewtopic.php?f=51&t=3025) (french or english): <http://forum.rallyesim.fr/viewtopic.php?f=51&t=3025>
- 2) [SISCO's forum](http://btbtracks-rbr.foroactivo.com/dudas-sobre-el-metodo-zaxxon-f37/) (spanish or english): <http://btbtracks-rbr.foroactivo.com/dudas-sobre-el-metodo-zaxxon-f37/>
- 3) [simracing's forum](http://foro.simracing.es/bobs-track-builder/4093-dudas-y-preguntas-del-ma-zaxxon.html) (spanish): <http://foro.simracing.es/bobs-track-builder/4093-dudas-y-preguntas-del-ma-zaxxon.html>
- 4) [devtrackteam](http://devtrackteam.solorally.it/viewtopic.php?f=28&t=69) (english): <http://devtrackteam.solorally.it/viewtopic.php?f=28&t=69>

XIII.3 Old help files

<http://www.multiupload.com/5B4KUL1ZO0>

or

<http://www.mediafire.com/?iuh4bp81hlnsf77>

XIII.4 parlesportes' site

French language site created by parlesportes with interesting stuff about BTB, 3DStudioMax and also about the zaxxon's method. Always has the most updated tutorials for the method.

<http://plpcreation.tonsite.biz/>

<http://plpcreation.tonsite.biz/index.php/btb/methode-zaxxon>

XIV LICENSE

Neither this tutorial, nor the method described, nor the scripts developed by the author are “free software”. Redistribution or commercial use of the tutorial, the method or the scripts are prohibited. Modifications are allowed only for personal noncommercial use.

The author accepts no responsibility for any damage or harm resulting from the use of this method.