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_sobject.m Creates a list of SObjects to be inserted by hand in the Venue.xml. **add_sobject(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
`> [mapeo]=textread('mapeo.txt','%f');`
`> x=2380.47;`
`> z=-2350.67;`
`> [longitud altura latitud]=BTB_a_coor(x,0,z,mapeo)`
	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 mallado_regular	**venue**

coor_a_btb.m
Returns the BTB coordinates of a point given the terrestrial coordinates
`> [mapeo]=textread('mapeo.txt','%f');`
`> longit= -73.67;`
`> latit= 41.47;`
`> [x1 y1 z1]=coor_a_BTB(longit,latit,elevation,mapeo)`
	base directory

corregir.m For a given road, compares the elevation profile assigned using dar_altura and that obtained from elevation data (from lamalla.mat), and changes the terrain elevation data (lamalla.mat) to fit the elevation profile set with dar_altura. corregir also accepts a kml file as a parameter and uses its coordinates and altitude to change lamalla.mat. This could be useful if we have a kml with altitudes we trust, but dangerous as those altitudes could have an offset respect to the elevation data available. corregir('file.kml')	**s3_road**
creartrack1.m Gets elevation values for a road from its coordinates and elevation data (lamalla.mat)	**s3_road**
create_hlg Creates file s1_mesh\sasplanet.hlg (open it with SASPlanet) with the boundary coordinates (box) of anchors_carretera.msh (run trocea_malla before using create_hlg)	**s1_mesh**
create_sons Creates the folder's structure for several sons in a multi-track project. **create_sons(number_of_sons)** Creates son01, son02, etc. folders in the same folder where the father is located **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	**father's root directory**
cut_lamalla.m Reduces the size of lamalla.mat. Useful if data comes for a too big zone. cut_lamalla([xmin xmax],[zmin zmax])	**s2_elevation** **s2_elevation_b**

dar_altura.m Softens the output from creartrack1 and gives the nodes of the track their elevation and slope to fit that curve. **dar_altura(smooth_factor,pos_slope,neg_slope,step,interactive)** - 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	**s3_road**
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.	**s1_mesh\salida**
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 *ideal* 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 gmsh.	**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 latittud 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**
leetif.m Creates lamalla.mat from a geotiff file	**s2_elevation** **s2_elevation_b**
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)	**s1_mesh**

make_grid.m
Creates several files containing a regular grid of points with terrestrial coordinates. Those files should be "raised" with BTBLofty 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 BTBLofty.
 	
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)`
	s2_elevation
	s2_elevation_b

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	**s1_mesh**

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)	**s7_walls_b**
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	**s10_split**
pacenotes.m Gets the track shape from a driveline.ini file. Output from thius 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 nc_splitted 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. Example: `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**
start.m 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 **start** **start(step,step_b,file_size)** Parameters are the steps in meters used by make_grid and the number of points per kml.	
terrain_noise.m 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 **terrain_noise([ymin ymax])**	**s4_terrain**
trocea_malla.m Splits anchors_carretera.msh into 2 parts: list of mesh nodes (nodos.txt) and triangles (elements.txt)	**s1_mesh**
vercontorno.m Shows a contour plot using the terrain elevation data (lamalla.mat) and the road position (output from btb06)	**s2_elevation** **s2_elevation_b**