

Topographic modeling for hydrological purposes using LiDAR originated data sources

Ioan Ferencik, 5.6.2010

The idea of using DEM for extracting hydrological data is not new, however it poses challenges when it comes to LiDAR data mainly because of size of data (millions of points) as well as the complexity and variability of resulting DTM/DEM.

Tools: libLAS, lastools, mcc-lidar, ILWIS, SAGA GIS, GRASS GIS, GDAL/OGR, Ubuntu OS

Requirements: at least 2GB RAM, 2GHZ processor(more cores the better).

Workflow:

1. LiDAR data usually comes preprocessed, in rectangular blocks in a projected coordinate system. Preprocessed means the point cloud is already classified. While this is a positive fact and the classification can be good for some purposes, it is not good enough for hydro data extraction. Therefore it is recommended to reprocess the data.
2. Multi Scale curvature classification is an algorithm developed by Evans and Hudak for forest filtering of LiDAR data and it was found to produce satisfactory results. However, at current time there is no perfect algorithm that works in flat areas, hilly areas, rural areas, urban areas, forests. Usually it is required to apply more than one algorithm.

mcc-lidar OPTIONS input-file output-file

Required options:

-s [--spacing] arg nominal post spacing (spatial resolution of analysis)

-t [--threshold] arg curvature threshold

Pass-related options:

-N [--nonground-pts] write non-ground points classified by each pass

-S [--surfaces] write raster surfaces interpolated by each pass

Other options:

--version display version

--help display help message

Example: **mcc-lidar -s 1 -t 0.2 input.las output.las**

The command accepts binary LAS file and outputs classified LAS file.

3. Extract ground points:
las2las -i input.las -o output.las -keep_class 2

also is possible to clip to a certain bounding box: **-clip 630250 4834500 630500 4834750**

4. Export to X;Y,Z,I (intensity)

las2txt -i input.las -o output.txt -parse xyz -sep 'komma'

5. create a GRASS location(use EPSG code)

- import the text file using v.in.ascii
v.in.ascii -z -b -t input=input.txt output=point_map fs=' ' x=1 y=2 z=3

-z create 3D vector
-t do not build table (it is slow) and not necessary
-b do not build topology (it is slow) and not necessary

- set the resolution and region for the interpolation

Note. Because we used -b flag the resulted vector map does not have topology, as such its bounds are unknown. So it is not possible to set the region from the map. As a result the region has to be entered manually. The bounds can be found either with **lasinfo** or by generating a boundary shapefile with **lasboundary (might be available only for Windows)**. Another option is to use las2shp and the ogrindex to generate a index shapefile, import in in grass using v.in.ogr and then set the region.

set the resolution (the same value as the -s in mcc-lidar)
g.region res=value

- Interpolation of LiDAR points is cumbersome and requires some expertise. Regularized Spline with Tension is a good interpolation program but it is resource intensive and requires some tuning on trial and error basis.

```
v.surf.rst --o input=sample_points tension=10 layer=0 elev=sample.rst smooth=0.1  
segmax=3 npmin=40 dmin=0.5 dmax=1
```

--o overwrite existing maps

input - vector points map (2d or 3D). if 3D than layer=0, else zcolumn parameter must be provided and layer=1

elev – resulting DEM

smooth=parameter specifying smoothness of the spline function. With smoothing parameter greater than zero the surface will not pass exactly through the data points and the higher the parameter the closer the surface will be to the trend

tension=surface with tension set too low behaves like a stiff steel plate and overshoots can appear in areas with rapid change of gradient and segmentation can be visible. Increase in tension should solve the problems. -t flag can be used to let the program normalize the tension and calculate a value that will be used in the interpolation mainly for numerical stability

segmax – number of segments in a block (RST is implemented using quad tree based segmentation). For LiDAR a small number is good due to high data density (4 for example). A high number will smooth too much the DEM

npmin – number of points outside the current segment at runtime used for approximating the values at the edges of the segment. This parameter is very tricky for LiDAR data. As a result of classification there will be holes in the point cloud. If these holes are bigger than the segment size, the segment inside the file will be flat (no points so the value of segment will be given from npmin or neighbor segment). A too high value will slow down the process and smooth the DEM producing high residuals. A too low value will generate blocky surface. Usually npmin will be deduced based on intuition and density of points in a trial and error process.

9. visualization of DEM in NVIZ
start nviz with nviz -q &
load the surfaces

QGIS is a very usefull tool with GRASS plugin. It is recoomended for map display and queries.

ILWIS or R can be used to generate a DEM suing Kriging. In this case first a semivariogram is built and a model selected and then surface is interpolated. Note that ILWIS will run out of memory because of LiDAR file size.