LiDAR data processing with SPDlib in QGIS.

R. Antolín[(1)], *P. Bunting*[(2)] *and J. Suárez*[(1)]

(1) Forest Research, Northern Research Station, Roslin EH25 9SY, UK.
{roberto.antolin,juan.suarez}@forestry.gsi.gov.uk
(2) Department of Geography and Earth Sciences, Aberystwyth University, Aberystwyth, Ceredigion, SY23 3DB, UK. pfb@aber.ac.uk

Resumen

La utilización de datos LiDAR (Light Detection And Ranging) es, hoy en día, la principal fuente de datos para la generación de modelos digitales de elevación. Hasta ahora no existía ningún paquete libre con garantías para el procesado de datos láser, sin embargo, la librería libre SDPlib suple esta carencia. Además de ser la implementación del formato SPD (Sorted Pulse Data), el paquete SPDlib es un conjunto de herramientas para la gestión, el procesado y el análisis de datos LiDAR de datos láser. Algunas de las prestaciones de la librería SPDlib son la conversión entre formatos LiDAR, clasificación de puntos, generación de modelos de elevación, y el cálculo de métricas estadísticas. La manera básica de utilizar las herramientas es por línea de comando, pero en la actualidad la librería se ha integrado dentro del cliente de escritorio QGIS. En este artículo, se presenta la librería SPDlib y su interfaz gráfica empotrada en el conjunto de herramientas Processing de QGIS mediante el caso práctico de elaboración de un MDE. Se analizarán las prestaciones que ofrecen estas herramientas frente a otros programas privativos.

Palabras Clave: LiDAR, QGIS, SPDlib, Jornadas, SIG, software libre, Girona.

Abstract

The use of LiDAR data (Light Detection And Ranging) has been increasing and is, today, the most commonly used data source for the generation of digital elevation models. Until now, there was no FOSS package able to process laser data with certain guarantees, however, SDPlib library supplies this lack. The SDPlib library is the implementation of the SPD format (Sorted Pulse Data) but also a set of tools for LiDAR data manipulation, processing and analysis. SPDlib is able to convert between different data formats, to classify the point cloud, to generate elevation models and to calculate statistical metrics. SPDlib tools are mainly used by command line, however, a new graphical interface has been integrated into QGIS desktop client. It is our intention to present here the SPDlib interface based on the QGIS Processing Toolbox and also to briefly show its use in order to produce basic cartography. A case of study is used to show the benefits that these tools may offer over other proprietary programs.

Key words: LiDAR, QGIS, SPDlib, Jornadas, SIG, FOSS, Girona.

1. Introduction

The use of **LiDAR** (Light Detection and Ranging) [23] for the generation of basic cartography is quite common nowadays. LiDAR is based on the survey of the territory by airborne laser scanners (**ALS**) or terrestrial laser scanners (**ALS**). Sensors measure the distance between the emission point and the surface being surveyed very accurately with high sampling rates. The final data from a LiDAR survey is a great amount of points (some hundred millions) that usually includes the planimetric coordinates of the points and their corresponding ellipsoidal heights. This amount of points are commonly known as *point cloud* which is a set of attributed points in a three-dimensional space. Thanks to the fact that LiDAR measurements are almost nadiral, laser rays are able to reach the ground surface, even through very narrow passages in highly vegetated zones. Digital Terrain Model (**DTM**s) are easily generated by means of an interpolation of ground points. Besides, Digital Surface Models (**DSM**s) represent the trend of the terrain and the objects attached to it, so they can be obtained as an interpolation of the whole point cloud. Other important LiDAR products are Canopy Height Models (**CHM**s) and Digital Building Models (**DBM**s), which basically consist in subtracting the DTM from the DSM. Thanks to these products, LiDAR applications go from terrain elevations and digital terrain modelling to 3D building [**Yu2010**, **Zhou2013**, 16] and man-made structures modelling [19], forestry [20, 25, 7, 5], power line mapping, engineering works [14], geology [18], hydrology [12], etc.

There are many proprietary and free software packages that can manage and analyse LiDAR point clouds. TerraScan [22] and TerraModeler [21] are the most widespread proprietary software. Amongst the GIS solutions, ArcGIS incorporates a module in order to process LiDAR data and recently it has developed its own proprietary laser data format. Vendors of laser equipments usually distribute their own software to pre-process raw LiDAR data, but in some cases this software is able to process and analyse data too. LAStools [13] is a very complete set of tools for LiDAR processing. Although they are freely distributed (as in *free beer*) for particular purposes, licensing is needed for commercial and government use.

LASlib [8] library, which is the free API used by LAStools, comes with writing and reading LiDAR data capabilities as well as some of the LAStools functionalities such as 3D viewing and format transformation. LASlib also incorporates the LASzip library [9] for LiDAR data compression. The libLAS library [15] supports writing and reading different laser data formats, but it also allows some data processing. PDAL [4] is a library to translate and manipulate a wide variety of data formats and it is analogous to the GDAL raster library. There are also GIS desktop clients with built-in modules intended for the visualization, as gvSIG, and the analysis, like GRASS [10]. In this last case, GRASS classifies points and interpolates them into digital elevation models [1].

Proprietary software licenses are commonly very expensive and changes in its functionalities are not permitted. On the other hand, those free software packages lack most of the functionalities needed for LiDAR processing. The SPDlib [3] software, however, overcomes these shortcomings. The SDPlib library is the implementation of the SPD format

(Sorted Pulse Data) but it is also one of the few free set of tools able to manage, process and analyse LiDAR data. The aim of this paper is to provide an overview of the SPDlib capabilities. We will first provide a description of the **SPD** file format, outlining how data is archived and what kind of information it stores. Secondly, we will show a basic work flow within the SPDLib framework which we will be described in detail. Finally, we will compare SPDlib with other privative software.

2. SPDlib

The SPDlib is one of the most complete FOSS libraries for LiDAR data manipulation, and it is able 1) to read and convert different types of data formats, both discrete (terrestrial and airborne LiDAR) and *full waveform* (see Section 2.1), 2) to decompose full waveform data into discrete pulses, 3) to select points by means of spatial and attribute queries, and 4) to divide point cloud points into tiles. SPDlib incorporates several ways to classify vegetation and terrain from the point cloud, and it also allows the creation of digital terrain, surface and canopy models using different interpolation methods. A very wide range of metrics can be computed from raw LiDAR thanks to its built-in templates and mathematical operators.

2.1. Sorted Pulse Data (SPD) format

The SPD file format has been designed specifically to store LiDAR waveform and discrete return data acquired by Terrestrial Laser Scanning (**TLS**), Airborne Laser Scanning (**ALS**) and space borne systems [2]. To provide better storage and access to data from pulsed laser systems, a new set of data structures is used. Data is stored within the Hierarchical Data Format version 5.0 (**HDF5**) file format [11]. Data can be stored both sorted or unsorted. Unsorted Pulse Data (**UPD**) does not impose any order on the data, which is usually time-sequential and therefore equivalent to the standard LASer (**LAS**) file format [17]. In contrast, Sorted Pulse Data (**SPD**) defines a spatial index which orders the pulses.

A pulse represents a single measurement unit from a pulsed laser system. The information associated with a pulse includes a list of points and two data blocks that store the transmitted and received waveform data. These waveforms are viewed as samples from one measurement.A points represent single discrete returns from the transmitted pulse. In SPD format pulses are stores as arrays and references a list of the individual returns within each pulse. Pulse parameters such as the origin coordinates, index point, absolute GPS time, wavelength and range of the received waveform from the origin are stored. Pulses also include a height field that can be populated independently with the height of the origin above the ground surface following its identification. Point fields are aligned with the LAS 1.x specifications [17] but additional parameters like the height of the point above the ground surface, the range of the point from the origin and the offset of the return within the waveform are included.

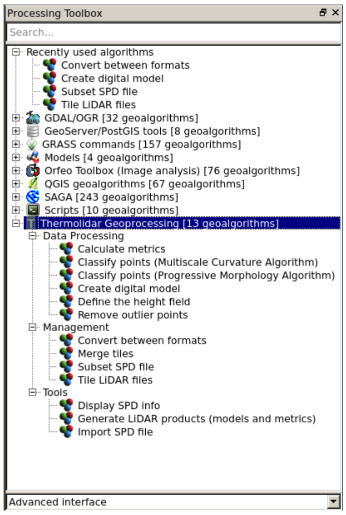


Figura 1: *Processing Toolbox with ThermoLiDAR plugin*

Representing data as a pulse-based format presents a number of advantages over point-based. Pulse-based format aligns with the sensor technology and the process of data acquisition, and it stores the information associated with each individual pulse. All pulses are stored, including those without any return. This is useful for applications such as quantifying vegetation canopy structure (e.g., gap fraction) and cover from TLS data. Returns are explicitly connected and ordered within the pulse which permits computationally efficient processing.

3. QGIS interface

SPDlib tools are command line based, however, some of their functionalities have been integrated into QGIS *Processing Toolbox* (Figure 1) as a plugin, providing a simple graphical user interface. To install the plugin the source code should be copied into the QGIS plugins folder. Through this section a description of the work flow will be used to briefly explain the different SPDLib tools and QGIS modules (Figure 2).

3.1. LiDAR workflow

LiDAR data is often provided as a number of tiles or flight lines. Depending on our computer capacity, in some cases it might be convenient to merge those files into a single file (*spdmerge*) and in others, to divide data in overlaping tiles with a more appropriate size (*spddeftiles* and *spdtiling*). LiDAR data is usually provided as LAS format files and it has to be converted to an indexed SPD file (*spdtranslate*). Noisy data can be eventually removed (*spdrmnoise*) before being processed. Once the SPD file has been created, ground points have to be classified (*spdpmfgrd* and *spdmccgrd*). Then the points height relative

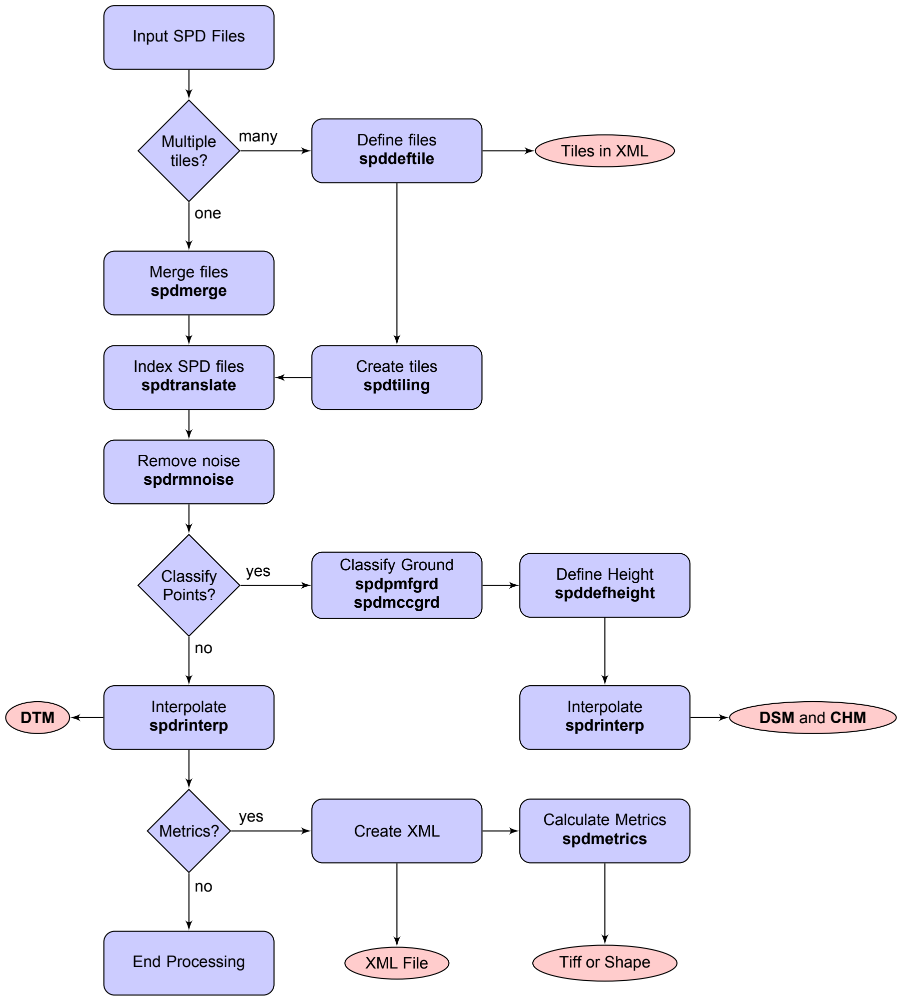


Figura 2: *LiDAR Workflow with SPDlib*

to the ground can be inferred (*spddefheight*). After defining the above-ground height field, ground and no-ground points can be interpolated (*spdinterp*) in order to obtain DTM, DSM and CHM. From height information a range of metrics (*spdmetrics*) commonly applied to LiDAR data and specifically to vegetation applications can be calculated.

3.1.1. Convert File Formats

spdtranslate converts between various supported file formats and coordinate system. In order to convert from LAS to SPD, input and output files and formats, and the index resolution and location are required:

```
$ spdtranslate -i Example.las --if LAS -o Example.spd --of SPD -b 1 -x
    FIRST_RETURN
```

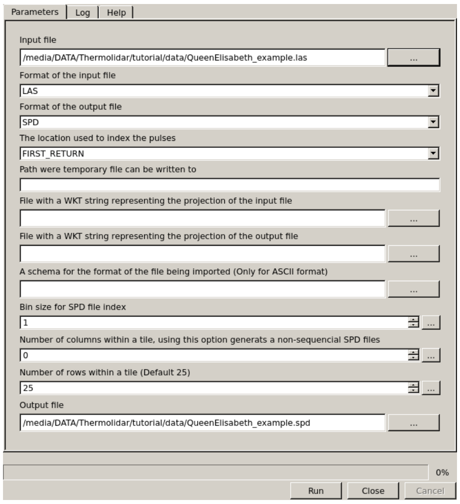


Figura 3: *spdtranslate GUI. Blank fields are optional.*

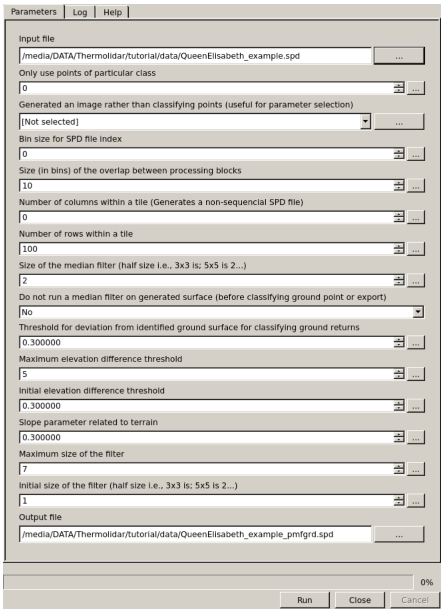
The formats supported are SPD/UPD, LAS and ASCII. The LAS reader is via libLAS [15] and therefore it only supports the discrete return (LAS 1.2) data. The ASCII format requires a schema, written in XML, to be supplied. The parse expects a single return per line and the resulting pulses will only contain a single return.

Input files are usually read into memory and sorted into a spatial grid before being written into the output file. In case of not enough memory the file needs to be tiled into blocks small enough to fit into memory. This is slower but once completed it is very fast to make spatial selections within the file and other processing steps can be applied to the whole file with only a relatively small memory footprint. In the Graphical User Interface (**GUI**) this corresponds to the number of rows and columns (Figure 3).

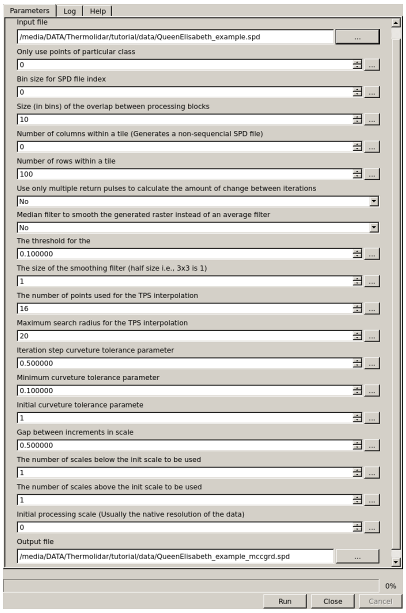
3.1.2. Classify ground returns

Although SPDlib has four different algorithms in order to classify ground points, only two of them have been implemented into QGIS: **spdpmfgrd** (Figure 4a) and **spdmccgrd** (Figure 4b). The spdpmfgrd command is an implementation of the progressive morphological filter (**PMF**) algorithm [24]. The algorithm works by generating an initial minimum return raster surface at the bin resolution of the SPD file and at each scale a morphological operation is performed in order to obtain ground points. The spdmccgrd command is an implementation of the multi-scale curvature (**MCC**) algorithm [6]. This algorithm classifies points exceeding positive surface curvature thresholds, resulting in a ground/no-ground classification. Under most circumstances the default parameters for both the algorithm will be fit the purpose and it is recommended to try the simplest commands:

```
$ # Apply PMF filter
$ spdpmfgrd -i Example.spd -o Example_pmfgrd.spd
$ # Apply MCC filter
$ spdmccgrd -i Example.spd -o Example_mccgrd.spd
```



(a) ***spdpmfgrd*** *interface*



(b) ***spdmccgrd*** *interface*

Figura 4: *Classification modules: PMF algorithm (a) and MCC algorithm (b). Except for the input and output file names, all the parameters are set to default values.*

3.1.3. Define Height Field

SPD files contain both elevations corresponding to a vertical datum and an above-ground height attributes for each discrete return. The **spddefheight** command is used to populate this height field. There are tow methods for doing this. A DTM with the same resolution than the SPD file indexation can be used as a reference surface. However, if the DTM is noisy, it will introduce undesirable effects. The other way is to interpolate a value for each point generating a temporal continuous surface, and so reducing any artefacts:

```
$ spddefheight --interp -i Example_mccgrd.spd -o Example_height.spd
```

The default interpolation is *natural neighbour* method. More about the interpolation methods and parameters are discussed in Section 3.1.4.

3.1.4. Interpolate DTM and DSM

spdinterp uses points classification (e.g. ground or no-ground) and height values (e.g. ellipsoidal or above ground heights) to create DSM, DTM and CHM. For instance, interpolating topographic elevations generates a DTM, and interpolating heights above ground generates a CHM. DSM are obtained by interpolating ground points:

```
$ spdinterp --dtm --topo -i Example_mccgrd.spd -o DTM.tif -b 1 -f GTiff
$ spdinterp --dsm --height -i Example_mccgrd.spd -o CHM.tif -b 1 -f GTiff
$ spdinterp --dsm --topo -i Example.spd -o DSM.tif -b 1 -f GTiff
```

Output raster resolution (`-b` option) and format (`-f`) are required. Several interpolation algorithms are available for calculating terrain models: a TIN interpolation, a Nearest Neighbour interpolation and a Natural Neighbour interpolation. The default interpolation is Natural Neighbour method.

3.2. Generate metrics

spdmetrics calculates metrics, which can be simple statistical moments, percentiles of the point height or return amplitude, or even count ratios. Mathematical operators can be applied to either other operators or metric primitives to allow a range of LiDAR metrics to be derived. Multiple metrics can be calculated at the same time if listed within an XML. This XML file has to be defined *a priori* with a hierarchical list of metrics and operators. Within the *metrics* tags a list of metrics can be provided by the *metric* tag. Within each metric the *field* attribute is used to name the raster band or vector attribute. Here is an example of an SPD metrics XML file to calculate the 95th percentile.

```
<spdlib:metrics xmlns:spdlib="http://www.spdlib.org/xml/">
    <spdlib:metric metric="percentileheight" field="95thPerH" percentile="95"
        return="All" \class="NotGrd" lowthreshold="0.1" />
</spdlib:metrics>
```

4. Comparison of SPDlib with other Lidar tools

In this section we compare SPDlib with other well known libraries. Comparison is made qualitatively by means of a table of functionalities and characteristics. In Table 1 four different software packages are presented: Fusion, LASTools, SPDlib and Terrascan. Fusion is very popular for Forest cartography production being its main characteristic the computation of LiDAR metrics. Fusion is free software (as in free beer) but its source code is closed. Fusion imports both LAS and ASCII formats. Because it has not developed since 2006, it does not support waveform data. LASTools is increasingly used in the last years. Only some of its modules are free software and cross-platform. The rest are closed and they can be used freely for not commercial purposes on Microsoft systems. LASTools supports 1.X LAS specifications, thus it does not support waveform data. LASTools generates any kind of digital model in both ASCII or GTIFF formats from raw LiDAR point clouds and calculates some height metrics too. Terrascan is the most widespread proprietary software for LiDAR processing, able to classify points and to generate digital models. Terrascan works with LAS and ASCII LiDAR formats only so waveform format is not supported.

SPDlib tools have been also quantitatively compared with Terrascan software by means of height differences between digital models. A dataset in Aberfoyle, Scotland, has been used for this purpose. The whole area has been classified both using SPDlib tools and Terrascan. Results have been interpolated in order to generate DTMs and CHMs. DTM from

Function	FUSION	LASTools	SPDLib	Terrascan
License	BSD	LGPL / Commercial	GPL3 / MIT-X	Commercial
Source Code	Closed	Mixed Open / Close	Open	Close
Import LAS/ASCII	Yes	Yes	Yes	Yes
Raster formats	ASCII / Binary	ASCII / GTIFF	GDAL	ASCII / Proprietary
Canopy Height Metrics	Yes	Yes	Yes	No
Point Cloud Z Metrics	No	No	Yes	No
Intensity Metrics	No	No	Yes	No
Metrics Calculator	No	No	Yes	No
Ground Classification	Yes	Yes	Yes	Yes
Merge Point Clouds	Yes	Yes	Yes	Yes
Interpolate Surfaces	Yes	Yes	Yes	Yes
Assign RGB Values	No	Yes	Yes	Yes
Tile Point Cloud	No	Yes	Yes	Yes
Mosaic Rasters	Yes	No	Yes	Yes
Data Indexing	No	Yes	Yes	Yes
Waveform Processing	No	No	Yes	No

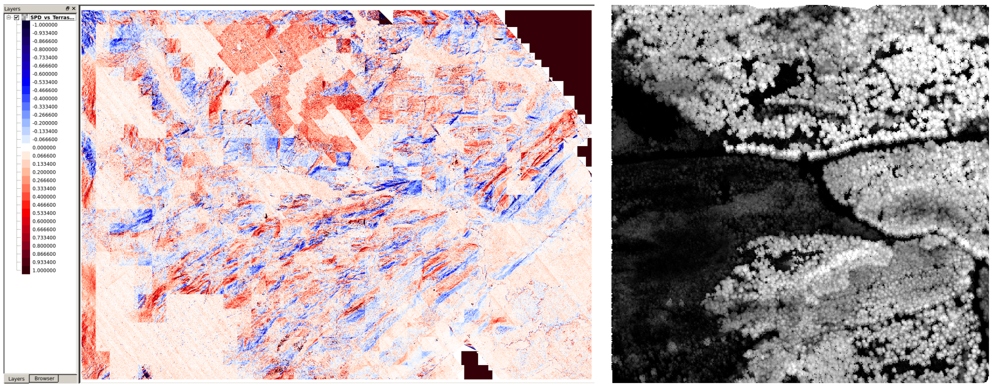
Cuadro 1: *Comparison of the functionality within SPDLib to FUSION, LASTools and Terrascan.*

SPDlib has been subtracted from DTM Terrascan and the map of differences is shown in Figure 5a. Tiles 500m$\times$500m wide are clearly recognizable within this map.

Tile *row13col16* has been chosen for a deeper study (see Figure 5) because it contains both positives and negatives differences. Within this tile, differences between vegetation points and CHMs, and ground point and DTMs are shown in Figure 6a and Figure 6b, respectively. Points classified as vegetation show a negative bias in both cases, being greater in the case of Terrascan ($\mu = -1{,}95$ m). Dispersions are equal to $\sigma = 3{,}1$ m. Averages are almost null for ground points, but Terrascan presents a greater value ($\mu = 0{,}09$ m). Dispersions are $\sigma = 0{,}13$ and $\sigma = 0{,}2$ for SPDlib and Terrascan, respectively. Whether interpolators overestimates heights in rough regions or filters tend commit commission errors (classify as ground points that belongs to vegetation) is not still clear, and further analyses are needed.

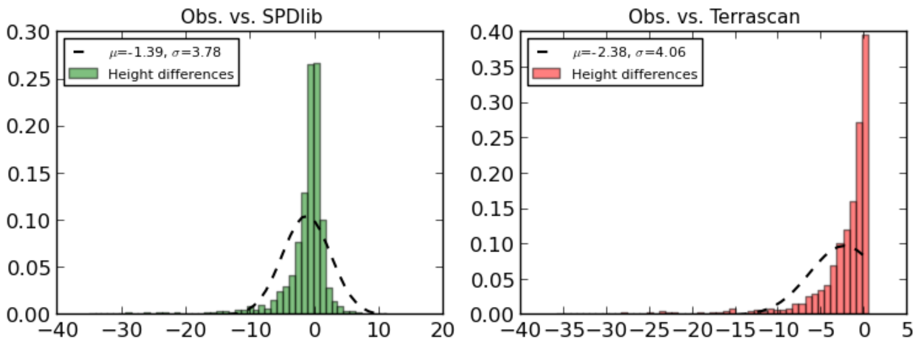
5. Conclusions

The capabilities and functionalities of SPDlib have been presented. The GUI within QGIS has been shown too. SPDlib has been compare with other LiDAR libraries and packages and a few results have been discussed. SPD is the most complete set of tools for LiDAR processing available nowadays. A basic study of height differences has revealed that SPDlib has performed the best results when creating a CHM. A deeper comparison of height differences as well as a better quality control of digital models obtained from SPDlib are needed in order to assess the real SPDlib performances. A more detailed description of how SPDlib point classification performs would be also necessary.

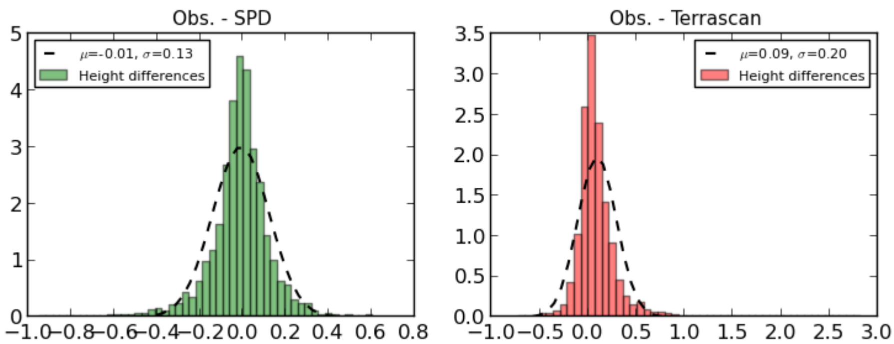


(a) **(b)**

Figura 5: *(a) Map of differences between SPDlib and Terrascan DTMs; Red colours mean negative differences and blue represent positive differences. (b) CHM of tile* row13col16*. It is characterised by a strong presence of high and medium vegetation*



(a) *Vegetation*



(b) *Ground*

Figura 6: *Height differences of vegetation points (a), and ground points (b).*

Acknowledgement

This work was supported by ThermoLiDAR project funded within the Seventh Framework Program (fp7-315165)

Referencias

[1] Roberto Antolín y Maria A. Brovelli. «LiDAR data filtering with GRASS GIS for the Determination of Digital Terrain Models». En: *I Jornadas de SIG Libre*. Girona, Spain, mar. de 2007, pág. 13. ISBN: 3-906467-69-4.

[2] Peter Bunting y col. «Sorted pulse data (SPD) library. Part I: A generic file format for LiDAR data from pulsed laser systems in terrestrial environments». En: *Computers & Geosciences* 56 (jul. de 2013), págs. 197-206. ISSN: 00983004. DOI: 10.1016/j.cageo.2013.01.019.

[3] Peter Bunting y col. «Sorted pulse data (SPD) library—Part II: A processing framework for LiDAR data from pulsed laser systems in terrestrial environments». En: *Computers & Geosciences* 56 (jul. de 2013), págs. 207-215. ISSN: 00983004. DOI: 10.1016/j.cageo.2013.01.010.

[4] Howard Butler y Michael Gerlek. *PDAL - Point Data Abstraction Library*. 2014. URL: http://www.pdal.io/index.html.

[5] J. Estornell y col. «Estimation of biomass and volume of shrub vegetation using LiDAR and spectral data in a Mediterranean environment». En: *Biomass and Bioenergy* 46 (nov. de 2012), págs. 710-721. ISSN: 09619534. DOI: 10.1016/j.biombioe.2012.06.023.

[6] Jeffrey S. Evans y Andrew T. Hudak. «A Multiscale Curvature Algorithm for Classifying Discrete Return LiDAR in Forested Environments». En: *IEEE Transactions on Geoscience and Remote Sensing* 45.4 (abr. de 2007), págs. 1029-1038. ISSN: 0196-2892. DOI: 10.1109/TGRS.2006.890412.

[7] António Ferraz y col. «3-D mapping of a multi-layered Mediterranean forest using ALS data». En: *Remote Sensing of Environment* 121 (jun. de 2012), págs. 210-223. ISSN: 00344257. DOI: 10.1016/j.rse.2012.01.020.

[8] rapidlasso GmbH. *LASlib*. 2014. URL: http://rapidlasso.com/.

[9] rapidlasso GmbH. *LASzip - free and lossless LiDAR compression*. 2014. URL: http://www.laszip.org/.

[10] GRASS Development Team. *Geographic Resources Analysis Support System (GRASS GIS) Software*. Open Source Geospatial Foundation. USA, 2012. URL: http://grass.osgeo.org.

[11] The HDF Group. *Hierarchical data format version 5*. 2000-2014. URL: http://www.hdfgroup.org/HDF5.

[12] Chengquan Huang y col. «Wetland inundation mapping and change monitoring using Landsat and airborne LiDAR data». En: *Remote Sensing of Environment* 141 (feb. de 2014), págs. 231-242. ISSN: 00344257. DOI: 10.1016/j.rse.2013.10.020.

[13] Martin Isenburg. *LAStools: software for rapid LiDAR processing*. 2014. URL: http://www.liblas.org/.

[14] M. Lato y col. «Rock bench: Establishing a common repository and standards for assessing rockmass characteristics using LiDAR and photogrammetry». En: *Computers & Geosciences* 50 (ene. de 2013), págs. 106-114. ISSN: 00983004. DOI: 10.1016/j.cageo.2012.06.014.

[15] libLAS Development Team. *libLAS - LAS 1.0/1.1/1.2 ASPRS LiDAR data translation toolset*. 2014. URL: http://www.liblas.org/.

[16] Niko Lukač y col. «Buildings roofs photovoltaic potential assessment based on LiDAR (Light Detection And Ranging) data». En: *Energy* (ene. de 2014). ISSN: 03605442. DOI: 10.1016/j.energy.2013.12.066.

[17] American Society for Photogrammetry & Remote Sensing. *LAS Especification Version 1.3*. 2010. URL: http://www.asprs.org/a/society/committees/standards/LAS%5C_1%5C_3%5C_r11.pdf.

[18] Joshua J. Roering y col. «'You are HERE': Connecting the dots with airborne lidar for geomorphic fieldwork». En: *Geomorphology* 200 (oct. de 2013), págs. 172-183. ISSN: 0169555X. DOI: 10.1016/j.geomorph.2013.04.009.

[19] George Sithole y George Vosselman. «Bridge detection in airborne laser scanner data». En: *ISPRS Journal of Photogrammetry and Remote Sensing* 61.1 (oct. de 2006), págs. 33-46. ISSN: 09242716. DOI: 10.1016/j.isprsjprs.2006.07.004.

[20] Juan C. Suárez y col. «Use of airborne LiDAR and aerial photography in the estimation of individual tree heights in forestry». En: *Computers & Geosciences* 31.2 (mar. de 2005), págs. 253-262. ISSN: 00983004. DOI: 10.1016/j.cageo.2004.09.015.

[21] TerraSolid. *TerraModeler – Surface Modeling and Terrain Mapping*. 2014. URL: http://www.terrasolid.com/products/terramodelerpage.html.

[22] TerraSolid. *TerraScan – Software for LiDAR Data Processing and 3D Vector Data Creation*. 2014. URL: http://www.terrasolid.com/products/terrascanpage.html.

[23] Aloysius Wehr y Uwe Lohr. «Airborne laser scanning—an introduction and overview». En: *ISPRS Journal of Photogrammetry and Remote Sensing* 54.2-3 (jul. de 1999), págs. 68-82. ISSN: 09242716. DOI: 10.1016/S0924-2716(99)00011-8.

[24] Keqi Zhang y col. «A progressive morphological filter for removing nonground measurements from airborne LIDAR data». English. En: *IEEE Transactions on Geoscience and Remote Sensing* 41.4 (abr. de 2003), págs. 872-882. ISSN: 0196-2892. DOI: 10.1109/TGRS.2003.810682.

[25] Kaiguang Zhao y col. «Characterizing forest canopy structure with lidar composite metrics and machine learning». En: *Remote Sensing of Environment* 115.8 (ago. de 2011), págs. 1978-1996. ISSN: 00344257. DOI: 10.1016/j.rse.2011.04.001.