

LiDAR data filtering with GRASS GIS for the determination of digital terrain models.

R. Antolín Sánchez ⁽¹⁾ y M. A. Brovelli ⁽¹⁾

⁽¹⁾ Politecnico di Milano – DIIAR – Polo Regionale di Como, via Vallegio 11 – 22100 Como, Italy.
roberto.antolin@polimi.it, maria.brovelli@polimi.it.

RESUMEN

En los últimos años, es cada vez más frecuente la utilización de todo tipo de Modelos Digitales. La tecnología LiDAR (Light Detection and Ranging), basada en la medición del territorio con un telémetro láser aerotransportado, permite la creación de Modelos de Superficie (Digital Surface Models, DSM) de manera sencilla mediante una simple interpolación de los datos. Sin embargo, los resultados más importantes que se pueden conseguir con esta tecnología son los Modelos Digitales del Terreno (Digital Terrain Models, DTM), donde han sido filtrados todos los objetos sobre el terreno, incluidos edificios, grandes estructuras (puentes, diques o pasos elevados), vegetación, etc. El Laboratorio de Geomática del Politécnico di Milano – Campus de Como- ha desarrollado una serie de algoritmos para, en un procedimiento automático, filtrar dichos datos LiDAR dentro del software GIS GRASS. GRASS se convierte en un programa óptimo para este tipo de proyectos, capaz de trabajar con una gran cantidad de puntos de una manera bastante rápida. Además, y dado que es un software de código abierto, ofrece la oportunidad al usuario de conocer aquello que realmente hace el algoritmo (imprescindible en un ambiente universitario).

Palabras clave: GRASS, LiDAR, software y de código abierto.

ABSTRACT

In the last years, the use of every type of Digital Elevation Models (DEMs) has improved. The LiDAR (Light Detection and Ranging) technology, based on the scansion of the territory by airborne laser telemeters, allows the construction of Digital Surface Models (DSM), in an easy way by a simple data interpolation. However, it is often necessary to produce more refined digital models, called Digital Terrain Models (DTMs) whose construction requires the identification and removal of the existing objects on the territory (vegetation, buildings, infrastructures). The filtering of such data has been carried out using specific algorithms implemented by the Laboratory of Geomatica of the Politecnico di Milano - Campus of Como -, suitably integrated by an automatic procedure to build DTMs within the GIS GRASS software. GRASS becomes a very useful tool for this kind of projects because it is able to process a huge amount of data and, since it is an open source software package, it offers the opportunity of knowing which are the computations the algorithm is actually doing.

Key words: GRASS, LiDAR, Free and Open Source Software

INTRODUCTION

The Airborne Laser Scanning (ALS) technology is based on the ground survey from an airborne laser telemeter. The telemeter measures the distance between the emission point, A, and the echoing point, B, which is a generic ground point hit by the laser ray. Thus, the laser telemeter measures the distance between the instrument and the echoing surface.

However, the ground point coordinates are actually wanted. The measure of these coordinates implies the knowledge of the airplane position and attitude at each instant. For this purpose, an integrated sensor GPS/INS (Global Positioning System/Inertial Navigation System) is provided.

This instrumentation basically consists of an inertial sensor which is composed of three accelerometers and three gyroscopes, a GPS receiver and an electronic device to synchronize and to archive the data of the instruments. The accelerometers and the gyroscopes are lead to measure the linear acceleration and angular velocity. Once the measuring session is over, the data is pre-processed by a Kalman filter to calculate the aeroplane position and attitude at each singular moment of the flight.

So, the GPS/INS sensor is able to determine the aircraft coordinates and its normal vector direction; The point distance from the telemeter and the angle between the emitted ray by the telemeter and the aircraft normal vector are also known. Thus, the coordinates of the surveyed point can be achieved.

Some of the most important laser scanning capabilities are:

1. high accurate measurements: 30 cm in planimetric components; and 15 cm in height component.
2. high resolution, (function of the height and velocity of the flight, and the scanning frequency) between 0.5 and 5 point/m².
3. high velocity survey. From a few km²/h up to 50 km²/h.

Due to two basic characteristics of LiDAR technology (monoscopy and almost-nadirality), laser rays are able to reach the ground or the studied object surface, also through very narrow passages in highly vegetated zones.

The final data from a LiDAR survey is a great amount of planimetric coordinates, sorted by the point's retrieved instant, and the corresponding ellipsoidal heights. Since LiDAR is often able to measure the intensity echo, this kind of signal attribute is also archived.

From LiDAR data, it is easy enough to develop a Digital Surface Model (DSM) as a simple raw data interpolation. DSM just represents the trend of the terrain and of the objects over it. However, the principal aim is to develop a Digital Terrain Model (DTM) or a Digital Building Model (DBM). Considering the height coordinates and using their geometry, the idea is to automatically classify the data. This process is named *filtering*, and the classification depends on each process. In general, this classification consists of two different feature types: terrain and object. Thus, filtering LiDAR data needs some algorithms to determine the points belonging to the terrain or to each single object. Those "object" points are removed before the DTM calculation.

FILTERING ALGORITHMS IN GRASS GIS

In the last years, some different DTM filtering algorithms have been developed. Usually, these algorithms are sensitive to the morphological characteristics of the DMSs and to the spatial density of the surveyed points. In 2002, the International Society of Photogrammetry and Remote Sensing (ISPRS) Working Group III/3 “3D Reconstruction from Airborne Laser Scanner and InSAR data” made the comparison between the different filtering algorithms, just to value their performance. This study identified the conditions in which the algorithms reproduce their best results.

Between these filters with best results, there were the Axelsson algorithm [3], based on the TIN (Triangulated Irregular Network) progressive densification; the Vosselman [19] and [20], Sithole [16] and Roggero [15] algorithms based on the mathematic morphology and the terrain slope analysis; the Sithole algorithm [18] based on the data segmentation, and finally, the Kraus and Pfeifer [14] algorithm. This last one calculates the DTM by a robust interpolation in which the non terrain points, considered as outliers, are removed.

However, in this paper the filter developed by the Laboratory of Geomatica at the Politecnico di Milano – Campus of Como – will be explained. This filter, also included in the ISPRS study, is based on the identification of object’s edges by a slope gradient analysis. These objects can be both buildings and vegetation. After the edge detection, their inside area is recognized and filled. Finally, and after the removal of these areas, the terrain surface is reckoned by an hybrid norm interpolation using bilinear or bicubic splines ([7], [8] and [9]).

Automatic classification

From the first beginning and due to educative purposes, our filtering algorithm was thought to be a free and open software. Thus, the GRASS (Geographic Resources Analysis Support System) project [13] was chosen to publish our packages in. There are some reasons for this. First of all, GRASS is probably the most powerful GIS of the “open” *community*, so there are a lot of available software libraries that make easier the implementation. Secondly, it is widely spread and this makes the algorithm usable by a lot of people. This is, of course, also an advantage, because it is possible to use the *community* resources.

The first versions of our filtering algorithm are available in GRASS 5.4. They were tested in different times with good results ([2] and [17]). However, in this paper we present the same modules into the new GRASS 6.X vector architecture [4]. Unfortunately, the command's porting could not be finish till the 6.2 version. This version contains all the LiDAR tools that will be used during this part.

The LiDAR filtering software developed by the Laboratory of Geomatica at the Politecnico di Milano within GRASS GIS 6.2 can be summarized by:

- ◆ Outlier removal.
- ◆ DSM interpolation by bicubic or bilinear splines, residual classification and buildings and vegetation edge detection.
- ◆ Determination of the area inside these edges.
- ◆ Building and vegetation removal and terrain surface reconstruction by bilinear spline interpolation with hybrid norm.

The execution of these steps is not carried out by an unique GRASS command able to filter a whole raw LiDAR data set. The GRASS commands implemented to carry out this whole procedure are, in order, *v.outlier*, *v.lidar.edgedetection*, *v.lidar.growing* and *v.lidar.correction*, although they do not exactly follow the same division as the points above. This fact allows the user to verify the data filtering step by step. Thus, it allows modifying its parameter each time for the best development of each DTM. There is a last command, *v.surf.bspline*, to generate a raster surface from a vector map by a spline interpolation. This command used to create the final DTM or a simply DSM will be deeply discussed in a forward section.

The dataset that will be used to illustrate how new commands work represents the Stuttgart's Railway Stations. Its UTM bounding coordinates are N=5403764, S=5403188, W=513118 and E=513632. The number of measured points is 259.030, for both last and first pulses. Thus, the mean resolution is about 0.87 p / m², due to some data gaps. This is one of the datasets used by ISPRS Working Group III/3 on the filter algorithms comparison. An aerial photo can be seen in figure 1.



Figure 1: Stuttgart Railway Station. © 2006 Google.

Outlier removal (*v.outlier*)

The first thing to do with any LiDAR dataset is a purge of the anomalous or outlier points within the data. These anomalous points are due to some different facts like gross errors in the point measurement, a passing bird during the measurement, clouds, etc. The basic idea to identify an outlier point is: if the value of a single observation differs a maximum threshold from a fixed value, reckoned from its surroundings by an interpolation, then it is considered as an outlier.

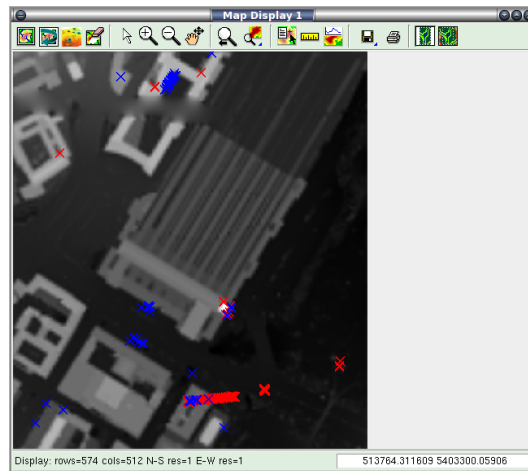


Figure 2: Outlier points. Red: 'First' points. Blue: 'Last' points.

The outlier identification is done by a bicubic spline interpolation of the observations with a high regularization parameter and a low resolution in south-north and east-west directions. In such a way, the interpolated surface is not sensible to the wrong or anomalous values. So, those points that differ in an absolute value more than a given threshold are removed. The default value is 50 (m), but in figure 2 a threshold of 25 m for the first pulse points and 30 m for the last pulse points were taken. As can be seen, although there are some point cluster that could create a bit confusion to the algorithm, this works pretty well identifying almost the totality of them. Of course, some outliers still remain and some of the points flagged as outlier are not so, for example, those on the station tower. But in this case, this last thing is not very important since the total station's structure is not modified and a correct recognition of its shape will be made by the next algorithm.

Objects and vegetation edges identification (*v.lidar.edgedetection*)

To detect the edge of each single feature over the terrain surface, two principles are used:

1. First of all, those points with a high gradient value are determined.
2. Then, it is valued the difference between the observation of those points and a generic interpolated surface. If the difference is positive, then the point could supposed to be inside the object, else the point could supposed to be outside the object.



Figure 3: *v.lidar.edgedetection* output. Yellow: 'Terrain' points; Red: 'Edge' points.

Figure 3 shows the object edges in red. The shapes of almost the total features in the landscape can be appreciated. This will permit the next command, *v.lidar.growing*, a good base for an optimal classification.

Determination of the inside contour area (v.lidar.growing)

A “region growing” algorithm is used to identify which is the internal area of every object. The input data is the output result of the *v.lidar.edgedetection* command. This new algorithm is based on the hypothesis that, in general, the heights inside are higher than on the contour. On the other hand, this procedure is much more reliable if the analysis is done with an interpolated surface because this layer is a smooth version of the studied object.



Figure 4: v.lidar.growing output. Yellow: 'Terrain single pulse' points; Green: 'Terrain double pulse' points; Blue: 'Object single pulse' points; Yellow: 'Object double pulse' points.

Before the application of the region growing algorithm, edges and their internal area must be known exactly. Because of that, another procedure must be done. A “convex hull” algorithm [10] is then applied. The purpose of this procedure is to identify the smaller convex polygon that contains every connected point forming an unique object.

In figure 4 the output of the *v.lidar.growing* is shown. The module is able to identify all objects in the landscape, also vegetation. Unfortunately, a procedure to distinguish between buildings and vegetation doesn't exist. This differentiation is essential in hydrological projects, where the volume of vegetation represented by this type of DSM (canopy), is not the same that the volume that actually obstructs the water flow (stem).

For understanding the figures 4 and 5 in a better way, colour code is explained. Yellow colour indicates points classified as “terrain” and corresponding to a “single pulse”. Red colour indicates points classified as “object” with a single pulse. The other two colours (blue and green) indicate an “double pulse object” point, and a “double pulse terrain” point, respectively. “Object double pulse” points generally mean edges, since the first pulse is the echo from the top of the object and the last pulse is the echo from its side. “Terrain double pulse” points can be considered as low and uniform vegetation zones due to the first and the last pulses coming from the top of the vegetation and from the ground, respectively. As said above, if the last pulse returned height is almost the same as the first pulse returned height, then the point is considered as “single pulse”. Instead, if last pulse returned height differs from the first one, the point is considered as “double pulse”.

Automatic correction of eventual errors in the classification (*v.lidar.correction*)

The automatic correction algorithm makes a comparison between the LiDAR observations and an interpolated surface in which only terrain points have been used. The best adapted surface to represent the terrain, also in presence of numerous and huge data gaps, is obtained with a bilinear spline interpolation and a small regularization value.

Once the new reference surface is obtained, which is a good approximation of the final DTM, two more steps should be done:

1. Points classified as terrain differing more than a threshold value are interpreted and reclassified as object.
2. Points classified as object and closed enough to the interpolated surface are interpreted and reclassified as terrain.

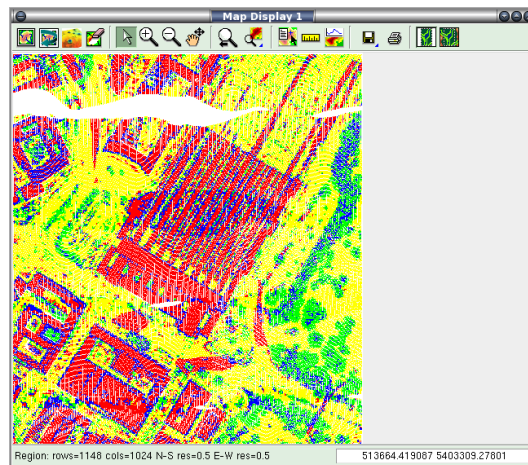


Figure 5: *v.lidar.correction* output. Yellow: 'Terrain single pulse' points; Green: 'Terrain double pulse' points; Blue: 'Object single pulse' points; Red: 'Object double pulse' points.

The figure 5 show the output result of the last command, *v.lidar.correction*. Unfortunately in this case, the correction procedure doesn't improve the *v.lidar.growing* results. This algorithm can be applied more than one single time to obtain a better final solution. But commonly, no better result is obtained after the second time.

Interpolation module: *v.surf.bspline*

The final DTM is created as an interpolation of the *v.lidar.correction* output where only 'Terrain single pulse' points are considered. Figure 6 shows the final DTM resulting after of the whole procedure.

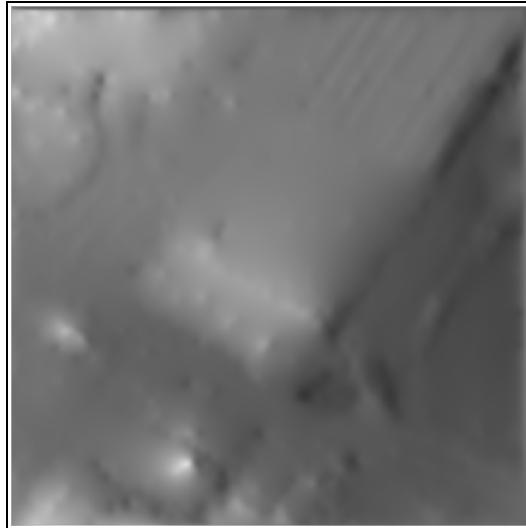


Figure 6: Final DTM

As already said, all commands shown till now dealing with LiDAR filtering work with bilinear and bicubic splines with Tychonov regularization. A least square system is considered in order to find out which are the best parameters, in a Markov sense, that fit the data. Due to possible lacks of data, causing a non invertible normal matrix, a Tychonov regularization is also considered. A gradient norm is taken in the bilinear case, and a curvature norm in the bicubic case. For a deep comprehension see [5].

The use of a least squares model to estimate the splines parameters causes the normal matrix to be enormous. This carries to the consumption of a lot of RAM resources and a great amount of execution time. From the first versions, the software was implemented in such a way to decrease those problems. First of all, optimized procedures were considered to make the normal matrix inversion faster. Also, the design matrix was optimized to avoid allocation memory problems. Secondly, the whole interpolating region is divided into smaller tiles. The number of subregions depends on the spline size in such a way that each tile has the same number of splines. Thus the number of normal matrices becomes greater but each subregion is also smaller with respect the whole region. Although there are more inversions to do, the needed time to do a single one is smaller, and thus, the total time for the whole interpolation is also smaller. To avoid border errors, an overlapping zone between neighbour tiles are considered [8]. In these overlapping regions, it is calculated a weighted mean with the height values from the involved tiles.

Unfortunately, importing modules to the new GRASS 6.X architecture changed the input reading in a considerable way. There were no more sites points but 3d vector points. Some tests using a PostGIS (PostgreSQL + GIS) database were carried out without any satisfactory result. First of all, it was tried to use the PostGIS geographic support. GRASS is able to work with it into two ways.

- a) **v.external** [12] allows to create a vector as read-only link to a OGR layer, in particular PostGIS.
- b) **v.in.db** [12] allows to import into GRASS any kind of database format supported (also PostGIS is available).

Thus, the only solution was considering the GRASS vector format. To simplify the input reading, a non optimal procedure was implemented. That is, each time a new tile was considered, the whole vector was read. This is a very important velocity problem

when LiDAR datasets are considered. A new approach was implemented into the interpolation module (*v.surf.bspline*) to solve this problem. It basically consists on grouping points related to a single tile. During the execution time, the vector map is read and site points are recorded into files accordingly to the tile that they belong. In this way, the vector map and the subregion files are read only one time.

This simple procedure was tested in three datasets with different number of point each one. Ordered by the number of points, the first dataset considered was the same as before, that is, the station of Stuttgart with 259.030 points. The second dataset with 1,507.468 points, is part of the LiDAR survey of the Como's Lake in the North of Italy. It represents the top of the lake and it is 1.3x2.1 Km² wide.

The last dataset is one of the 668 zones of the LiDAR survey of Po's River developed by the Authority of the Po river basin (AdbPo) [1]. This region is 2x2 Km² wide and contains 5,040.602 points. A little village called Portalbera, in the Pavia Region in the North of Italy, is within this zone. This dataset contains a lot of particularities like a urban landscape, vegetation mixed with buildings, man-made structures like dikes and also some lacks of data due to river. So, it is a magnificent dataset not only for testing LiDAR filtering but also how our interpolation module works with data gaps. In fact, some tests were made in that direction.

The test consists on a simple bilinear interpolation with different spline steps. The lower the step, the higher the number of tiles. The default spline step is 4 m but for these tests, some multiples and sub-multiples of the default value were considered, that is, 0.5, 1, 2, 4 ,8 and 16.

Table 1: Execution times for dataset 1 ('Stuttgart').

STUTTGART				
spline step	number regions (NSxEW)	old	new	gain
0.5	10x9	28m15s	27m39s	2,1%
1	5x4	5m46s	5m42s	1,2%
2	3x2	1m24s	1m24s	0%
4	1x1	19,37s	21,35s	-9,9%

Table 2: Execution times for dataset 2 ('Como').

COMO				
spline step	number regions (NSxEW)	old	new	gain
1	11x19	1h58m26s	1h14m24s	37,2%
2	5x8	14m56s	13m01s	12,8%
4	3x4	3m41s	3m18s	10,4%
8	2x2	1m01s	1m04s	-4,9%
16	1x1	13,12s	24,28s	-85,1%

Table 3: Execution times for dataset 3 ('Portalbera').

PORTALBERA				
spline step	number regions (NSxEW)	old	new	gain
1	16x16	2h16m42s	1h33m47s	31,4%
2	8x8	32m33s	22m18s	31,5%
4	4x4	8m03s	6m10s	23,4%
8	2x2	2m18s	2m30s	-8,7%
16	1x1	1m01s	1m41s	-65,6%

The results shown in the tables 1, 2 and 3 were obtained with a 3 GHz CPU and 2 Gb RAM computer. As expected, in cases where the whole region is considered in a single tile, the approach of files makes the interpolation slower. This is perfectly correct since if files are used, then points are read, written into files and read again and interpolated; while in the old procedure, points are read and directly interpolated. As it can be seen, in datasets without a high number of points (see Stuttgart's dataset), there isn't a gain in the execution time. This can be explained in the way that GRASS GIS 6.X has an optimized reading procedure. GRASS reading algorithm is fast enough to compensate the fact of reading several times the same vector map.

There is not any really advantage in the smaller dataset, Stuttgart, where the best performance is only a 2% faster. In Como's dataset, a gain in time of about 12% can be reached with a spline width of 1m. A very interesting improvement of 37% in the execution time is reached when a high number of tiles are considered into the interpolation. This is not really used since the spline resolution is greater than the point resolution (2p/m²). But anyway, this spline step is interesting because it shows which is the algorithm's behavior when some tiles have no data like in this case, where part of the Como's Lake is into the dataset.

But, as expected, a very good performance of the new implementation is obtained in the dataset with the highest number of points, Portalbera. Near a 30% on time execution is gained with respect to the old approach. At the same time, and also as the other datasets, with a low number of subdivisions, the new implementation doesn't have a good performance.

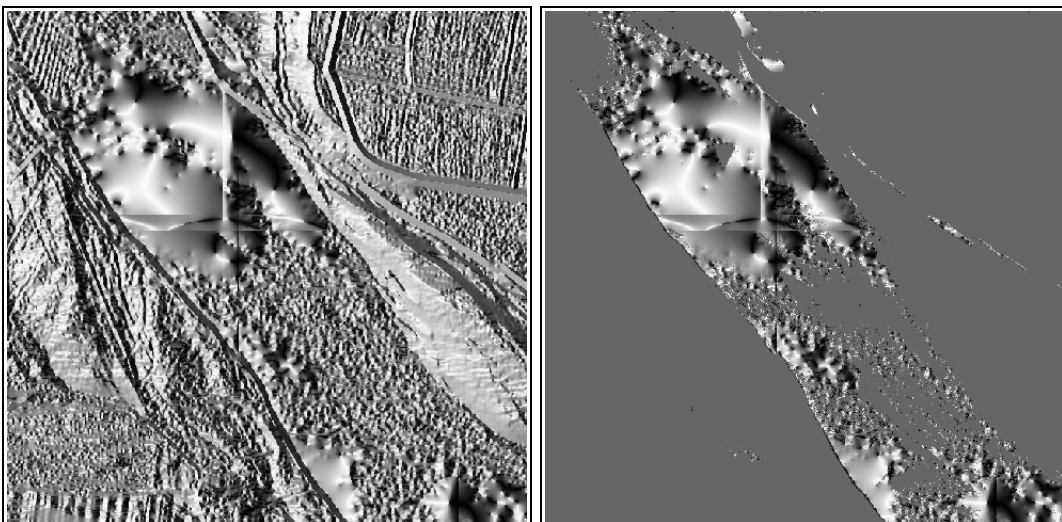


Figure 7: Interpolation problems. Horizontal and vertical bizarre features can be seen in the aspect map (left). The same aspect map with observation points rendered in grey (right).

There is a known problem of special interest in this interpolation module, when aspect (*r.slope.aspect*) or shaded (*r.shaded.relief*) maps are reckoned from a *v.surf.bspline* raster output. As said above, a weighted mean is calculated in overlapping regions. Due to the fact that this overlap involves in general only two tiles, the algorithm used in this regions is a linear mean. This makes that the borders of the overlapping zones have opposite slope directions (see figure 8). But it is only appreciated in parts where a big data gap is present as in figure 7, or in low density points zones. Those effects are produced due to very different interpolation trends in border zones.

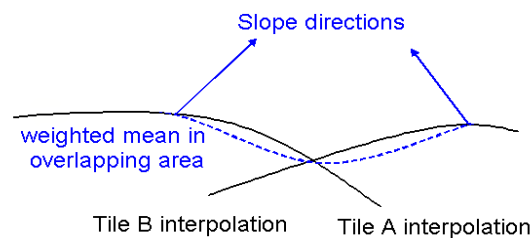


Figure 8: Interpolation problems. Horizontal and vertical bizarre features can be seen in the aspect map (left). The same aspect map with observation points rendered in grey (right).

CONCLUSIONS AND FUTURE WORK

New LiDAR tools are available in GRASS 6.2. They were ported from the older versions of the software, in particular from GRASS 5.4. This paper explains the filter procedure implemented by the Geomatic Laboratory and also show how commands still works quite well with the new GRASS 6.X architecture. Also the visualization of the results is faster and easier now with respect older versions since it is possible to use the new graphic user interfaces. So, GRASS GIS 6 becomes a magnificent software to work with LiDAR data.

A new reading implementation within the interpolation module, *v.surf.bspline*, has been carried out for a faster executions time with respect to current available version. An improvement of about 30% can be obtain in a vector map with approximately 5 million point. This new procedure is still advantageous with vector maps of about a million point and not so with small vectors. This makes the new implementation actually useful for faster results when huge vector maps are used. That is exactly the case of LiDAR data, where datasets with more than 10 millions points are very common and interpolation procedures are generally slow. However, the disadvantage of this approach is that a bit more than the vector size is needed as free hard disk space. But nowadays, powerful computers with some hundreds of gigabytes are very common.

Finally, an algorithm to automatically distinguish between building and vegetation must be done. This distinction, although it is useless for the transformation from a DSM to a DTM, is fundamental for hydraulic studies. The obstruction produced in the LiDAR echo by a tree canopy, is larger than its stem. This stem actually produces the

obstruction of the water flow. In the case of a building, however, its volume for the water obstruction is the same than the LiDAR surveyed volume.

The possible solutions for this problem are linked with the study of the shape of both object types: buildings usually show more regularity in their shape; vegetation is smaller and much more irregular. In complex situations, that is, vegetation placed against buildings, this distinction is not enough, so it would be necessary some extra information. This will be the aim of our further researches.

REFERENCES

- [1] AUTHORITY OF THE PO RIVER BASIN (2005). "Rilievi 'Laser-scan' del fiume Po da confluenza Ticino all'incile, comune di Ariano nel Polesine: specifica tecnica delle attività".
- [2] ANTOLÍN, R., BROVELLI, M.A., (2006). "Digital terrain models determination by LiDAR technology: Po Basin experimentation". *Bolletino di Geodesia e Scienze Affini, Anno LXV, n. 2. (In edition)*.
- [3] AXELSSON P., (1999). "Processing of laser scanner data - algorithms and applications". *ISPRS Journal of Photogrammetry & Remote Sensing, 54. pp. 138 -147*.
- [4] BLAZEK, R., NETELER, M. AND MICARELLI R., (2002). "The new GRASS 5.1 vector architecture". *Proceedings of the Open source GIS – GRASS users conference, Trento, Italy*.
- [5] BROVELLI M.A., REGUZZONI M., SANSÒ F., AND VENUTI G., (2001). "Modelli matematici del terreno per mezzo di interpolatori a spline". *Bolletino della Sifet*.
- [6] BROVELLI M.A., REGUZZONI M., SANSÒ F., VENUTI G., (2001). Procedure di ricostruzione del modello digitale del terreno da dati laser scanning. *Bolletino della Sifet*.
- [7] BROVELLI M.A., (2002). "Filtraggio di dati laser per la rimozione di forme". *Proceedings of the meeting "La tecnica del Laser Scanning: teoria ed applicazioni"*, Udine.
- [8] BROVELLI M.A. AND LONGONI U.M., (2003). "Software per il filtraggio di dati LiDAR". *Rivista dell'Agenzia del Territorio, n 3*.
- [9] BROVELLI M.A. AND CANNATA M.A., LONGONI U.M., (2004). "LiDAR data filtering and DTM interpolation within GRASS". *Transaction in GIS, Blackwell Publishing Ltd*.
- [10] DE BERG M., VAN KREVELD M., OVERMARS M., SCHWARZKOPF O., (2000). Computational Geometry: algorithms and applications. Utrecht, *The Netherlands. Springer Verlag*.
- [11] GRASS DEVELOPMENT TEAM, (2005). "Geographic Resources Analysis Support System (GRASS) Programmer's Manual". *ITC-irst, Trento, Italy. Electronic document: <http://grass.itc.it/devel/index.php>*.
- [12] GRASS DEVELOPMENT TEAM, (2005). "GRASS 6.2 Users Manual". ITC-irst Trento, Italy. *Electronic document: http://grass.itc.it/grass62/manuals/html62_user/index.html*.
- [13] GRASS DEVELOPMENT TEAM, (2006). "Geographic Resources Analysis Support System (GRASS) Software". *ITC-irst, Trento, Italy. <http://grass.itc.it>*.
- [14] KRAUS K. AND PFEIFER N., (2001). "Advanced DTM generation from LiDAR data". *IAPRS, Vol. XXXIV – 3/W4 Annapolis, MD, 22-24 Oct, pp. 23-30*.
- [15] ROGGERO, M., (2001). "Airborne Laser Scanning: Clustering in raw data". *IAPRS, Vol XXXIV – 3/W4 Annapolis, MD, 22-24 Oct, pp. 227-232*.
- [16] SITHOLE G., (2001). "Filtering of laser altimetry data using a slope adaptive filter". *IAPRS, Vol. XXXIV – 3/W4 Annapolis, MD, 22-24 Oct, pp. 203-210*.

- [17]SITHOLE, G., AND VOSELMAN, G., (2003). "Report: ISPRS Comparison of Filters ". *Electronic document*: <http://www.itc.nl/isprswgIII-3/filtertest/MainDoc.htm>.
- [18]SITHOLE, G., (2005). "Segmentation and Classification of Airborne Laser Scanner Data". *Publication of Geodesy, Netherlands Geodetic Commission. Delft, Netherlands*.
- [19]VOSELMAN, G., (2000). "Slope based filtering of laser altimetry data". *IAPRS, Vol XXXIII, Part B3, Amsterdam, The Netherlands. pp. 935-942*.
- [20]VOSELMAN, G., (2002). "Filtering of Laser Altimetry Data". *Proceedings of the meeting "La tecnica del Laser Scanning: teoria ed applicazioni", Udine*.