# SOLID IMAGE EXTRACTION FROM LIDAR POINT CLOUDS

D. Munaretto [a], M. Roggero [b]

[a] dafnemunaretto@gmail.com
[b] Politecnico di Torino, Dipartimento di Architettura e Design, Torino, marco.roggero@polito.it

**KEY WORDS:** point clouds, laser scanner, solid image

**ABSTRACT:**

In laser scanner architectural surveying it is necessary to extract orthogonal projections from the tridimensional model, plans, elevations and cross sections. The paper presents the workflow of architectural drawings production from laser scans, focusing on the orthogonal projection of the point cloud on solid images, in order to avoid the time consuming surface modeling, when it is not strictly necessary. The proposed procedures have been implemented in *fortran90* and included in the *VELOCE* software package, then tested and applied to the case study of the San Pietro church in Porto Venere (SP), integrating the architectural surveying with an existing bathymetric and coastal surveying.

## 1. INTRODUCTION

In laser scanner architectural surveying it is necessary to extract orthogonal projections from the tridimensional model, plans, elevations and cross sections. The paper presents the workflow of architectural drawings production from laser scans. Because of the surface modeling take a considerable amount of time (about the 30% of the total workflow) and a huge computational load, we propose to extract the architectural drawings directly from point clouds, avoiding the surface modeling step. Moreover the surface modeling generally produce a loss of the finest details, that in this way can be retained.

The geometric procedures to extract the orthogonal projections of point clouds will be defined. The products of these projections however are not traditional drawings, but are in the form of solid images. The solid image (or 3D image) is a concept introduced in photogrammetry by Prof. S. Dequal in 2003. While in a traditional bi-dimensional digital color image, consisting in a 3 band pixel-matrix, the RGB information is stored, in a 3D image a fourth band is required to store the third dimension, for example the distance from the section plane. A fifth band has also been used to store the surface reflectivity measured by the laser scanner.

The algorithms have been implemented in *fortran90* and included in the *VELOCE[1]* software package (Manzino & Roggero, 2002). *VELOCE* is intended for batch processing of large data sets. The software born in 2000 to process large airborne LIDAR data sets, and includes algorithms for point cloud filtering and segmentation. Later it has been adapted to terrestrial laser scans, including several functions for point cloud management, filtering and projection. *VELOCE* is actually available under GNU license for academic users, but new personalized functions can be implemented also for commercial users.

The studied procedures have been applied to the case study of the San Pietro church in Porto Venere (SP), integrating the architectural surveying with an existing bathymetric and coastal surveying, realized by Istituto Idrografico della Marina Militare and Codevintec in 2006. The survey consists of 15 scans in the interior and of 20 scans in the exterior, for a total of 743 millions of scanned points. The survey campaign required two days, while the post processing of the scanned points has required 3-4 weeks to filter and colorize the data, to align the point clouds and to produce the solid images of plans, elevations, cross sections and projection of the vaults. Finally it has been produced a video of the scanned point cloud, representing the San Pietro promontory, that is intended to be used by the Comune di Porto Venere for touristic promotion and divulgation purposes.

## 2. POINT CLOUD PROCESSING

In laser scan surveying production the largest amount of time is required by point cloud processing. This phase usually takes the 60÷80% of time of the entire work, while the surveying campaign is much less time consuming. This leads to an optimization of the entire surveying workflow, reducing the cost of the campaign and moving the human resources to the post processing phase.
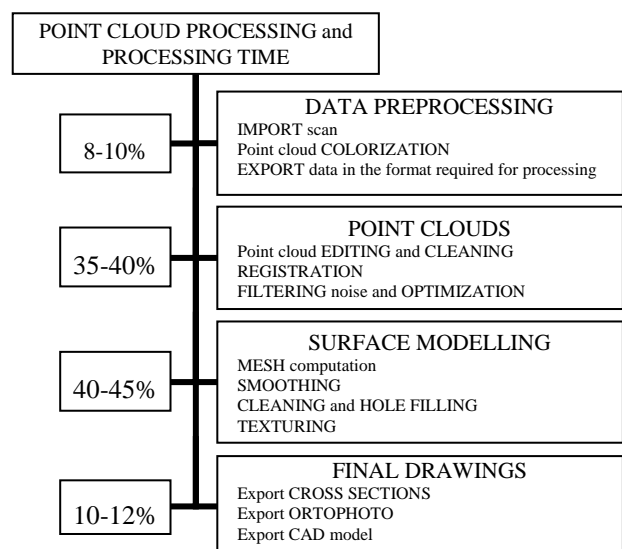


Figure 1. Point cloud processing times

The point cloud processing includes all the software operations that are needed to fit the customer requirements (i.e. textured

---

[1]

3D model, solid images, ortophoto, drawings, plans, elevations and cross sections, etc.). The workflow required in Fig. 1 roughly shows the processing time required by the different processing phases. The mesh computation with the necessary cleaning, hole filling, smoothing and texturing, takes about 40÷45% of processing time.

Because of all the geometric and radiometric information are already contained in the point cloud, we can avoid surface modeling, when the 3D model is not explicitly required by the customer. On the other hand, point clouds are not suitable for a quick extraction of vector drawings; for this reason the most of commercial software require surface modeling in order to extract cross sections. The present work will suggest an alternative way, that we have applied to architectural surveying.

## 3. POINT CLOUD PROJECTION ON SOLID IMAGES

The three-dimensional model (mesh surface or point cloud) is used to extract the final products, that usually can be vector drawings or ortophotos. Because of the traditional ortophoto of a point cloud is simply an orthogonal projection of the colored points onto a plane, its limit is that the third dimension is lost, while using the concept of solid ortophoto (Bornaz, Dequal, & Lingua, 2006), the third dimension is paired to the radiometric information: a fourth band is coupled to the three RGB bands, with the same spatial resolution, in order to store the distance of the points from a reference surface.



Figure 2. Solid image navigation

VELOCE produces 6 band solid images with this band structure:

- The R, G, B color components are stored in the $1^{st}$, $2^{nd}$ and $3^{rd}$ band, as short unsigned integers (1 byte per band).
- The laser intensity is stored in the $4^{th}$ band as short unsigned integers (1 byte).
- The distance from the projection plane is stored in the $5^{ft}$ band, as long signed integer (2 byte).
- A $6^{th}$ band can be used to record the number of scanned point projected in the pixel (scan density); to store this information 1 byte is enough.

The memory used by this data structure is 7 bytes per pixel.

In the case of horizontal cross sections (plans, projection of vaults) the third dimension can be the orthometric height of points, while in the case of vertical cross sections or elevations, it can simply be the distance from the projection plane.

Navigating the solid image the three dimensional coordinates of the points can be visualized or used to measure distances or angles[2]. An example is given in Figure 2.

### 3.1 Point cloud pre-processing and bounding box definition

We suppose to use registered and filtered point clouds, in cartesian coordinates x, y and z. The bounding box will be defined by

$$\begin{cases} x_{min} \\ y_{min} \\ z_{min} \end{cases} \text{ and } \begin{cases} x_{max} \\ y_{max} \\ z_{max} \end{cases}$$

The VELOCE software can import binary or ASCII files in .pts or .ptx format. The data are archived in binary format designed in order to have the maximum data compression and a fast access to the information, as described in (Manzino, Roggero, 2002). Because of the coordinates are stored as REAL[3] (4 byte), a shift $(x_T, y_T, 0)$ is applied to the E and N components in order to contain the values inside the limit range; in the following example:

| 1 | 566686,615 | 4877559,614 | 73,502 | 90 | 69 | 63 | 63 |
| 2 | 566686,614 | 4877559,613 | 73,503 | 382 | 74 | 71 | 72 |
| 3 | 566686,614 | 4877559,619 | 73,502 | -39 | 68 | 60 | 58 |
| 4 | 566686,615 | 4877559,616 | 73,500 | 25 | 68 | 63 | 61 |
| 5 | 566686,610 | 4877559,617 | 73,504 | 220 | 72 | 65 | 66 |
| 6 | 566686,623 | 4877559,611 | 73,502 | 446 | 79 | 69 | 77 |
| . | ... | ... | ... | ... | ... | ... | ... |

Tab. 1. Raw data imported from *.pts format file

| 1 | 0,000 | 0,000 | 73,502 | 90 | 69 | 63 | 63 |
| 2 | -0,001 | -0,001 | 73,503 | 382 | 74 | 71 | 72 |
| 3 | -0,001 | 0,005 | 73,502 | -39 | 68 | 60 | 58 |
| 4 | 0,000 | 0,002 | 73,500 | 25 | 68 | 63 | 61 |
| 5 | -0,005 | 0,003 | 73,504 | 220 | 72 | 65 | 66 |
| 6 | 0,008 | -0,003 | 73,502 | 446 | 79 | 69 | 77 |
| . | ... | ... | ... | ... | ... | ... | ... |

Tab. 2. Raw data in the VELOCE *.raw format
(binary file, after horizontal shift)

When the data are exported, the opposite shift $(-x_T, -y_T, 0)$ is applied to compute the original cartographic coordinates.

The VELOCE raw binary format includes some header record, where are recorded the number of points, the bounding box and horizontal shift.

The solid image is initialized with the following values:

- The background color is assigned to the R, G, B bands ($1^{st}$, $2^{nd}$ and $3^{rd}$ band)
- The value 0 is assigned to the $4^{th}$ band, the laser intensity
- To the $5^{ft}$ band, the distance from the projection plane, is assigned the maximum value of a long signed integer (32767). The distances will be then recorded in cm, up to the limit of 327.67 m, that is enough for the architectural applications.
- To the $6^{th}$ band is assigned the value zero.

### 3.2 Cross sections

Most of commercial software compute the cross section as the intersection between the mesh surface and a plane, as in Figure 3.

---

[2] The VELOCE software exports solid images in binary format, that can be visualized by ENVI.
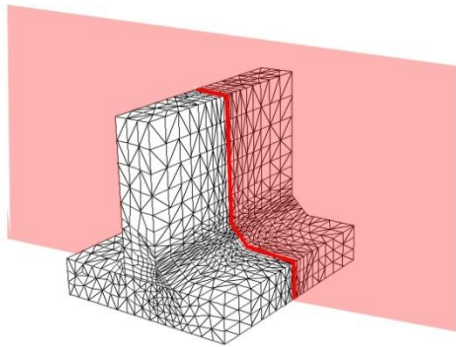
[3] floating point IEC 559 (IEEE)

Figure 3. Cross section from the mesh surface model

Because of the point cloud is, by definition, a set of adimensional entities, it is not possible to intersect it with a plane, but it is possible to find the points within a given distance $dz$ from the intersecting plane itself, as in Figure 4.
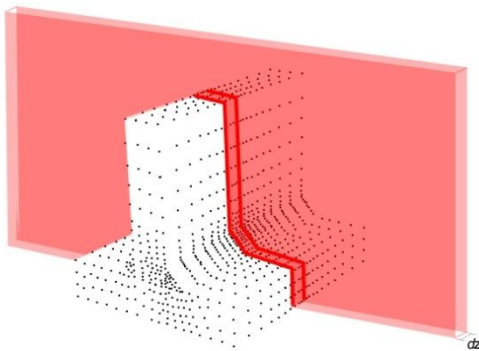


Figure 4. Cross section from the point cloud

The points in the interval $dz$ will be considered as coplanar and used to define the cross section line. The interval $dz$ must be defined by the user, and it depends on the point density. When a point it is find to be in the interval $dz$, the corresponding pixel on the solid image is colored in red (or in black or in another user defined color), while when a point is out of the $dz$ interval, it is projected with its proper color.
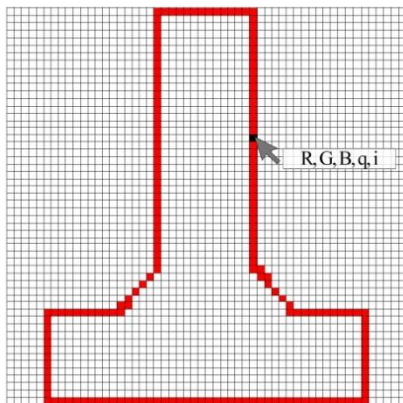


Figure 5. Discretized cross section projected from the point cloud on the solid image

We have seen as the distance band is initialized at the value 32767 (327.67 m), then the scanned points are sequentially red from the raw file and projected onto the solid image, with the following criteria:

- If the solid image pixel is void, the point is projected on the pixel

- If the solid image pixel is not void and the distance of the new point is lower than the distance in the $5^{th}$ band, the point is projected on the pixel

When the algorithm finds non hidden points to be projected, the pixel values in the solid image are updated, as shown in the following example of a plans generated by an horizontal section at the height of 1.5 m. Consider this points that are projected in the same image pixel:

| id. | x [mm] | y [mm] | z [mm] | ρ [-] | R [-] | G [-] | B [-] |
|-----|--------|--------|--------|-------|-------|-------|-------|
| 56 | 2546 | 3789 | -1277 | -1535 | 36 | 24 | 33 |
| 102 | 2540 | 3781 | -3 | -1503 | 38 | 23 | 32 |
| 388 | 2541 | 3782 | -200 | -479 | 59 | 50 | 48 |
| 5735 | 2545 | 3786 | 32 | 2033 | 117 | 11 | 114 |
| 8290 | 2545 | 3785 | 1735 | 1121 | 96 | 89 | 83 |
| 9006 | 2549 | 3785 | 1876 | 113 | 73 | 66 | 61 |
| 9197 | 2543 | 3788 | 1498 | 929 | 90 | 84 | 80 |

The algorithm read the point 56 and verify that is under (or behind) the section plane:

- $z_{56} < 1.50$ m
- $z_{56} > -327.67$ m

This is verified, so the value $z_{56} = -1277$ is written in the solid image and the 6 bands are updated as follows:

R(36) G(24) B(33) ρ(-1535) z(-1277) n(1)

Than the point 10 is red and

- $z_{102} = -3$ mm
- $z_{102} > z_{56}$

So the solid image pixel is updated and overwritten as follows:

R(38) G(23) B(32) ρ(-1503) z(-3) n(2).

The point 388 is skipped because the condition $z_{388} > z_{102}$ it is not verified. The point 5735 is accepted and used, while the points 8290 and 9006 are skipped, because the condition $z_{8920}$, $z_{9006} < 1500$ mm it is not verified. Finally the point 9197 is used and overwritten to the previous values as follows:

R(90) G(84) B(80) ρ(929) z(1498) n(3).

Fixing $dz = 50$ mm, the point 9197 is in the interval

$1500 > z > 1500\text{-}dz = 1450$

so the pixel must be colored in red and the RGB values updated:

R(255) G(0) B(0) ρ(929) z(1498) n(3).

The solid image is recursively updated up to the end of the raw data file.

The position of a pixel in the multi-band image is given by the pointer

$$p_{band} = \left[(n_{band}- 1)\cdot n_l \cdot n_c\right] + (l_i - 1) \cdot n_c + c_i$$

where

- $n_{band}$ is the number of the band (i.e. 1-6)
- $l_i$ is the pixel line
- $c_i$ is the pixel column
- $n_l$ is the total number of lines
- $n_c$ is the total number of columns

This algorithm is quite slow, requiring a sequential reading of the raw data file and an a priori unknown number of image pixels updates. Because of the points are registered by the scanner in a sequential order following the scanning path, it is possible to speed up the process neglecting some points when they can be considered not necessary (hidden points). We use the following algorithm to skip a number $j$ of points:

- If the point is behind the cross section plane, then $j$ is zero. In this case all the following points will be sequentially red and processed

- If the point is ahead the cross section plane it is not processed, and the probability is as larger as the dis-

tance from the plane is large, so we can skip some points. The number $j$ of point to skip can be

$$ j = \frac{d}{|\Delta d|} $$

where d is the distance from the cross section plane and $\Delta d$ the increment in this distance.

In any case the number $j$ must be limited to a maximum (i.e. $j_{max}$ = 500) in order to avoid to go back behind the cross section plane too rapidly, loosing non hidden points. The Figure 6 exemplifies how the algorithm works. Suppose to produce an horizontal section at the height $z_{sez}$. The algorithm read the point 1; it is ahead the section plane, so it must not be projected and probably we can skip some points. It is necessary to read the point 2 to compute $\Delta d$ and $j$; in the example we skip 201 points and jump to the point 204. It is now necessary to know how the scanning path is approaching the section plane, recomputing $\Delta d$ and $j$, so we need the point 205. The method allows to skip a large number of points, avoiding a lot of slow READ instruction. Only when the scanned points approach the section plane or when we are behind the plane, it is necessary to process all the points.
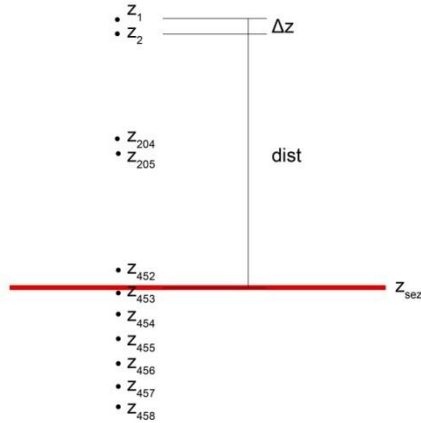


Figure 6. Raw data sequential processing and skipped points

The algorithm is analogous both for horizontal and vertical cross section (as in the example in Figure 2). In the case of vertical cross sections or elevations, it can be useful define partial or broken out sections. Also a vertical section can be orthogonally project on plans with different orientation, as shown in Figure 7, where a vertical cross section is defined by the horizontal polyline on the (x, y) plane. The order of the polyline vertex defines the orientation of the view, conventionally on the left following the polyline path.
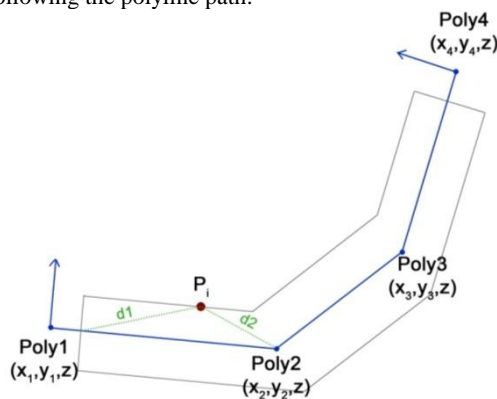


Figure 7. Plan of the projection on a broken vertical section plane: the cross section is in the middle line of a building.

In the case of vertical cross sections the coordinates of a point (x, y, z) are projected in the point $(x_p, y_p, z)$, belonging to the projection plane. If the vertical section plane is broken, also with different planes orientations, the first question is on which plane the point must be projected, the second one is the computation of the projected horizontal coordinates $(x_p, y_p)$.
At first we compute the horizontal distance between the point $P_i$ and the polyline vertex, Poly1, Poly2, …, PolyN, then we look for the two nearest vertex (Poly1 and Poly2 in the example). These two vertex identify the vertical plane on which the point $P_i$ must be projected.
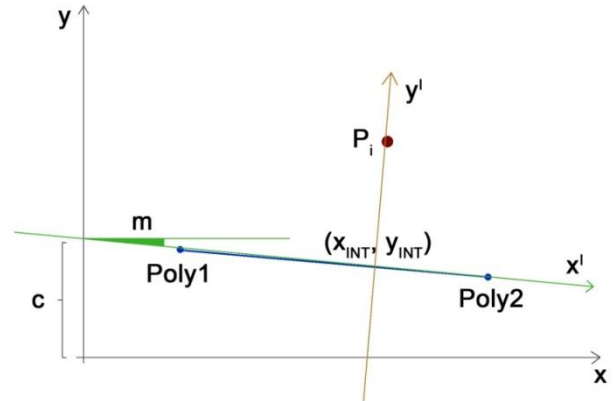


Figure 8. Projection of a point on a segment

The Figure 8 shows how to compute the horizontal coordinates of the projected point. It's easy to write, in the form y = mx + c, the equation of the line passing through Poly1 and Poly2 (tangent line T), and of the line passing through $P_i$ and normal to the segment $\overline{Poly1Poly2}$ (normal line N). The coordinates of the projection are given by the intersection between these two lines. The angular coefficients are given by

$$ m_T = \frac{y_{Poly2} - y_{Poly1}}{x_{Poly2} - x_{Poly1}} $$

$$ m_N = -\frac{1}{m_T} $$

and the two constant terms by

$$ c_T = y_{Poly1} - m_T x_{Poly1} $$
$$ c_N = y_P - m_N x_P $$

the horizontal coordinates of the projected point follows:

$$ x_{INT} = -\frac{c_T - c_N}{m_N - m_T} $$
$$ y_{INT} = m_N x_{INT} + c_N $$

The distance of the point $P_i$ from the projection plane (or from the projection point) cannot be simply computed as a module using the Pitagora theorem, because it is necessary to determine if the point is behind or ahead the projection plane. It is necessary to define a new coordinate system $(x^I, y^I)$, with origin in $(x_{INT}, y_{INT})$ and orientation of the $y^I$ axis toward $P_i$. the coordinates of $P_i$ in this new system are given by the transformation

$$ \begin{bmatrix} x_P^I \\ y_P^I \end{bmatrix} = R \cdot \begin{bmatrix} x_P - x_{INT} \\ y_P - y_{INT} \end{bmatrix} $$

where R is the rotation matrix given by

$$ R = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix} \qquad \alpha = \arctan(m) $$

While $x_P^I = 0$ by definition, $y_P^I$ represents the distance of $P_i$ from the projection plane, and will be stored in the band 5 of the solid image. Finally the position of the point in the image must be computed:

$$n_{l,P} = \left\lceil \frac{z_{max} - z_P}{res} \right\rceil + 1$$
$$n_{s,P} = \frac{x_f}{res} + 1$$

where:

$n_{l,P}$    Number of line of the point $P_i$
$n_{c,P}$    Number of column of the point $P_i$
$z_{max}$    maximum z value of the *bounding-box*
$z_P$    coordinate z of the point $P_i$
$x_f$    Progressive coordinate along the polyline
res    Image resolution (side of the pixel)

### 3.3 Tools for image enhancement

The obtained solid image obviously presents some lack of data, imperfections due to picture exposure and projections of distant or hidden objects. These shortcomings can be repaired to some extent, in order to make a more acceptable and readable image.
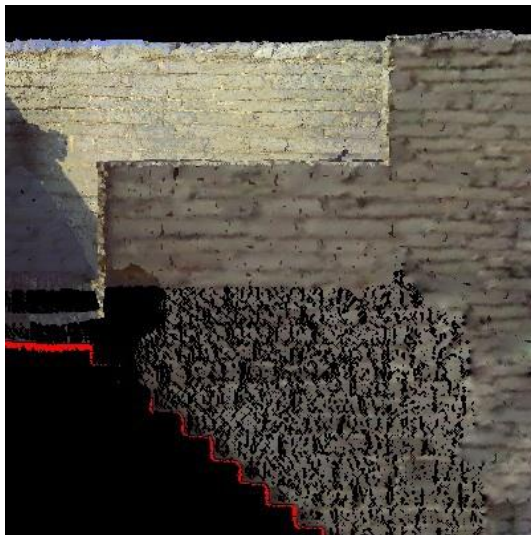


Figure 9. Solid image with lack of data filled

Lack of data in the point cloud can produce non colored pixels or pixels erroneously colored by projecting hidden points. Both cases are shown in Figure 2, and depends on the point density with respect to the image resolution. To correct for lack of data the VELOCE software scans the entire image with a 3x3 search matrix, loading both the RGB and the distance information. At first on the 9 elements of the search matrix it is computed the minimum distance $d_{min}$, then the pixel for which $d > d_{min} + (2 \div 3) \cdot res$ are considered hidden points and canceled out. If at this stage the central pixel of the search matrix is void, it is filled with a mean value of the other non void pixels, as shown in Figure 10.
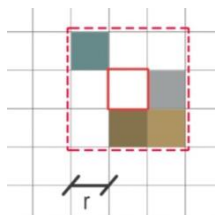


Figure 10. Search matrix used to correct for lack of data. The search radius r is normally set to 1.

The central pixel is filled only if a minimum of point in the search matrix is colored, in this case we use a threshold of 20%. The result is shown in Figure 9.

To improve the readability of the image it is also possible to use color mapping, in order to represent the point that are far from the projection plane with a modified brightness or tone. VELOCE allows to fix a distance $z_1$ from the section plane beyond which the points are represented by a modified color, and a maximum distance $z_2$ to completely exclude far objects. These distance intervals are represented in Figure 11. Color mapping can also be used in topographic surveying, to represent the ground heights in a DTM.
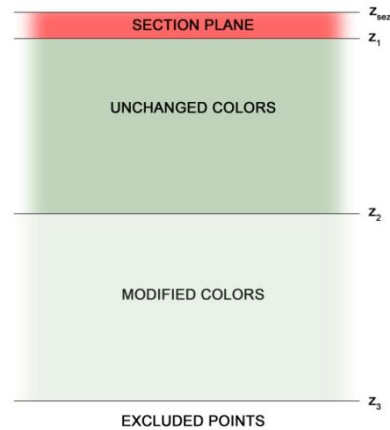


Figure 11. Distance intervals from the section plane; between dz and $z_1$ the point are represented in real colors, between $z_1$ and $z_2$ are represented in modified colors.

In the interval $z_1 \div z_2$ the color is modified in function of the parameter

$$d = \frac{z_1 - z}{z_1 - z_2}$$

that ranges from 0 to 1. The color RGB is transformed in the HSL system (*Hue*, *Saturation*, *Lightness*), so it is possible to modify one of these three components, in function of the desired graphic effect. For example, if we want to vanish the far points into a black background, we can modify the lightness by

$$L = L (1 - d)$$

In this way the lightness remain unchanged for $z = z_1$, and it is null for $z = z_2$. Finally the RGB components are updated.

Another tool for image enhancement has been developed to correct overexposures. Photographic images acquired by an integrated camera are commonly used by laser scanners to colorize the scan points. The integrated camera exposure is automatically set by the scanner firmware over a mean vale for the spherical panorama. This results obviously in some overexposed images, in the direction of lights, windows or sun. The Figure 12 shows a scan of the ceiling of a tunnel, where around the light projectors the image is clearly overexposed. Image exposure has been corrected by substituting the lightness L with the laser intensity; then the RGB values have been updated obtaining a much more readable color image.
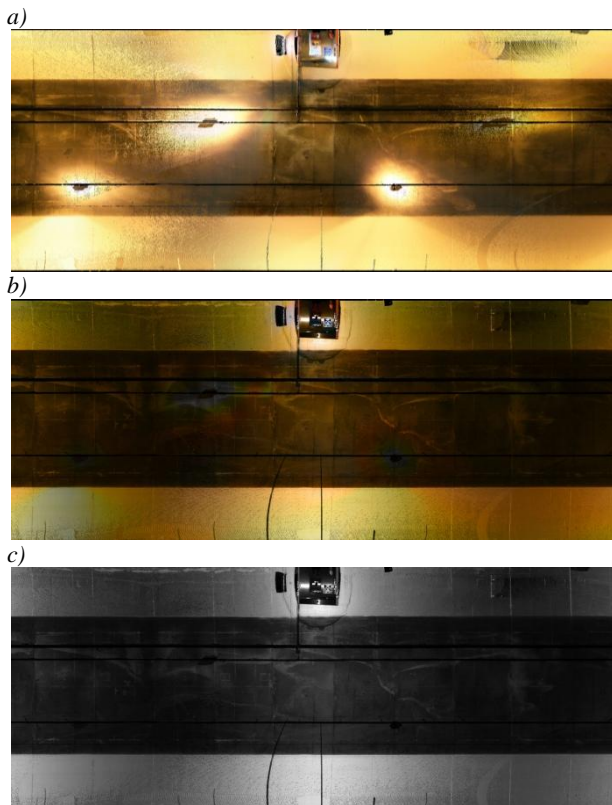
*a)*

*b)*

*c)*

Figure 12. Scan of the ceiling of a tunnel in cylindrical projection. The RGB image (in *a*) is overexposed around the light projectors, while the laser intensity (in *c*) is perfectly readable. In the *b* image the RGB lightness is tuned according to the laser intensity.

## 4.    THE CASE STUDY: THE SAN PIETRO CHURCH IN PORTO VENERE (SP)

### 4.1    The survey campaign

The VELOCE software has been used in a large number of aerial surveys, but to test it on terrestrial scans we used the case study of the San Pietro church in Porto Venere (SP), where it has been possible to acquire laser scans of the architectural object and of the surroundings. In the same area we had the data of the survey carried out by Codevintec[4] in collaboration with Istituto Idrografico della Marina Militare[5], that covers the entire area of the San Pietro promontory and of the seabed of the Porto Venere bay, between the seaside and the Palmaria island. A survey campaign on the church has been done by Codevintec in 2006 using an ILRIS 3D terrestrial laser scanner, while in 2008 it has been done the bathymetric and coastal survey, by an ILRIS MC (motion compensation) laser scanner and a SWATHPlus-H interferometric multibeam. During our campaign in 2012, the survey of the church has been completely redone by a FARO CAM2 Laser Scanner Focus[3D], acquiring 15 scans inside the church (with resolution of 6 mm at 10 m) and 20 scans outside the church (with resolution of 3 mm at 10 m); the apparently higher resolution of the outdoor scans is due to an higher scan distance. Merging the new survey with the previous surveys, we obtained a georeferenced point cloud of about

---

[4] http://www.codevintec.it/
[5] http://www.marina.difesa.it/conosciamoci/comandienti/scientif
ici/idrografico/Pagine/home.aspx

743 millions of points. For many commercial software it is almost impossible to process such amount of data in a single block. VELOCE can manage huge amounts of data that are limited only by the hard disk capacity, with quite acceptable processing time. We have tested the software on a laptop equipped by an Intel Core[TM] i7 and 8 GB of RAM. The point cloud has been converted in the RAW binary format in about 40 minutes, while the rendering of a single solid image of about 100 megapixels takes 15÷30 minutes.

### 4.2    Results

As example some results are shown. Figure 13 represents the raw solid image of a cross section exported by VELOCE. In this case the image have been enhanced repairing it only for lack of data, while color saturation and lightness have been leaved unchanged.



Figure 13. The raw solid image exported by VELOCE



Figure 14. The solid image after the graphic editing.

The raw solid image has been then edited by a photo editing software; it is notable that the three RGB bands of the solid image can be treated as a common photographic image. Editing involved the removal of hidden points and the inclusion of a pattern.

## 5. CONCLUSIONS

The goal that we set for this study, namely the extraction of drawings directly from the point cloud, has been fully achieved: while following the classical canons of the architectural drawings, we were able to preserve, through the solid image, the three-dimensional content that is intrinsic in laser scanner surveys.

This does not mean that the work is done here, on the contrary, have emerged still several areas to work on. It is still necessary to improve some aspects of the solid image graphics, the exposure tuning and the vectorization of the section line.

## 6. REFERENCES

Lingua, A., Agosto, E., & Del Bianco, P., 2007. L'ortofoto solida di precisione a grandissima scala in ambito urbano. In: *XI conferenza nazionale ASITA,* (vol. 2, pp. 1373-1378), Torino, Italy, ISBN 9788890313202

Bornaz, L., Dequal, S. & Lingua, A., 2006. L'ortofoto solida di precisione: un prodotto innovativo per la rappresentazione e la gestione dei dati tridimensionali. In: *Atti Conferenza Nazionale SIFET,* Palermo, Italy, ISBN 8890193921

Bornaz, L. & Dequal, S., 2003. A new concept: the solid image. In: *International Archives CIPA* N°XIX-2003 Vol. 1 ISPRS Vol. XXXVI-5/C34. CIPA 2003 XIXth International Symposium. Antalia (Turchia). 30 settembre – 04 ottobre 2003. (vol. 1, pp. 169-174). ISBN/ISSN: 975-561-245-9 / 0256-1840.

Manzino, A. & Roggero, M., 2002. Il trattamento informatico del dato laser. In: *La tecnica del laser scanning, teoria ed applicazioni,* pp. 39-74, Udine, Italy, ISBN: 888513727X

Roggero, M., 2000. Scansione laser da piattaforma aerea. Trattamento e analisi dei dati, thesis.