RECONSTRUCTING CCTV, BEIJING

Valentina Forcella ([a]), Luigi Mussio ([b])

DIIAR – Sezione Infrastrutture Viarie
Politecnico di Milano
Piazza Leonardo da Vinci, 32 – 20133 Milano
Tel. +39.02.23996605 ([a]) - +39.02.23996501 ([b]) – fax. +39.02.23996602
e-mails: valentina.forcella@virgilio.it ([a]) – luigi.mussio@polimi.it ([b])

KEY WORDS: Clustering, sowns and chains, Delaunay triangulation and the Bézier spline.

ABSTRACT:

This paper deals with the reconstruction of a building, starting from a point cloud. The shape of this building is a non-stellar concave and multi-connected structure, composed of sowns and chains. A sown is the representation of a horizontal plane formed by dense points. A chain is a planar loop modeled by rare points. CCTV structure is defined only by the three orthogonal Cartesian coordinates. The reconstruction uses a sequence of procedures and the desired output is a consistent 3D model. The first procedure is devoted to attributing points to their voxel and to estimating the three values needed afterwards. The second procedure is devoted to analyzing clusters vertically and horizontally, to preliminarily distinguishing chains from sowns and to generating relational matching. The third procedure is devoted to building closed loops between all chains and all their projections on sowns. The fourth procedure is devoted to connecting points with triangles. The fifth procedure, still being implemented, is devoted to interpolating triangles with triangular splines.

1. INTRODUCTION

This paper deals with the reconstruction of a building, starting from a point cloud. The test example is the China Central Television tower, located in Beijing.
The construction began in September 2004 on the 20 hectare site of an abandoned motorcycle factory in Beijing's new Central Business District and was completed by Office for Metropolitan Architecture (OMA) in August 2008.
The shape of this building is a non-stellar, concave and multi–connected structure, composed of sowns and vertical or almost vertical walls (treated as chains).
This structure is modeled by points, acquired from pictures available on the web and in architectural literature, and defined only by their three orthogonal Cartesian coordinates, without any other distinction, as shown in the following image, extracted by the data set.

1	0.000	0.000	-20.000
2	10.000	0.000	-20.000
3	20.000	0.000	-20.000
4	30.000	0.000	-20.000
5	40.000	0.000	-20.000
6	50.000	0.000	-20.000
7	60.000	0.000	-20.000

Figure 1. Extract of input data set

The imposition of an arbitrary reference system is needed to remove the uncertainty of the origin, orientation and scales.
The origin is placed on the left hand side, the axes as right-hand coordinate system and 1 m in reality is 1 unit in the data set.

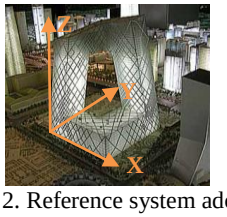


Figure 2. Reference system adopted

The results will be given in the same reference system in which the coordinates are provided.
The first 3D plot shows the structure represented by the point cloud.

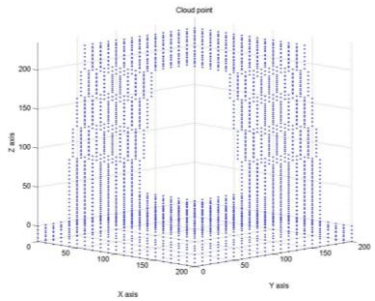


Figure 3. 3D plot of point cloud

3D reconstruction uses a sequence of procedures because it is almost impossible to use a single algorithm. The 3D model could be done in different ways, here the authors present the way chosen.

2. VOXEL

A voxel (volumetric picture element) is an element of the total volume, representing a value on a regular grid in three-dimensional space. The position of a voxel is inferred based upon its position relative to the other voxels.
The first procedure is devoted to checking the data set, to detecting the minimums and the maximums of X, Y and Z, to attributing points to their voxel and to estimating the three values needed afterwards.
The input file is CCTV.dat containing a list with the:

- ID point;
- X-coordinate;
- Y-coordinate;
- Z-coordinate.

The data set must be checked because different points could have the same coordinates or different points in different coordinates could have the same number ID. If one of these two cases occurs, a list with the number ID, coordinate X, Y and Z is provided. If there are no double points, it is shown. The service file CCTV_RID.dat contains the checked data which will be used subsequently.
In this specific case there are no double points or points out of height so CCTV.dat is equal to CCTV_RID.dat.
It is also necessary to find minimums and maximums for each coordinate, in order to set the voxel parameters.
In this specific case, the values given are:

- X minimum: 0.000;
- X maximum: 20.000;
- Y minimum: 0.000;
- Y maximum: 20.000;
- Z minimum: -20.000;
- Z maximum: 235.000;

For each order of voxel, a list provides:

- the total number of points;
- the voxel ID;
- the number of points that belong to that voxel.

Another list contains only the full voxels, their full-voxel ID, their voxel ID and the number of points that belong to that full voxel.
The "voxel" procedure continues up to the order in which the voxels are all empty.
In this specific case, the results are:

- voxel 1° has no point;
- voxel 2° has no point;
- voxel 3° has 1 point (in yellow in the picture below);
- voxel 4° has 167 points (in blue in the picture below);
- voxel 5° has 1077 points (in red in the picture below);
- voxel 6° has 1802 points (in green in the picture below).

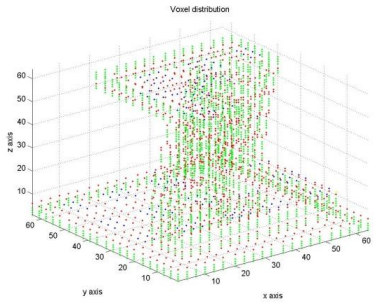


Figure 4. Voxel distribution

At the end of this procedure, the optimal order voxel is found with its amplitude in X, Y and Z .
In this specific case, the parameters produced are:

- Optimal voxel: 6°;
- X amplited: 3.125;
- Y amplited: 3.125;
- Z amplited: 3.984.

The "voxel" procedure is also used due to the estimatation of the three values needed afterwards.

- The first parameter defines vertical scanning and divides horizontal surfaces (which model sowns) from chains.
- The second parameter defines horizontal clustering and generates finite point sets at the same level, if necessary.
- The third parameter is used to match vertically between points belonging to consecutive levels; this parameter is also used to distinguish sowns from chains.

In CCTV reconstruction, the parameters produced are:

- height step: 7.969;
- planimetrical distance: 17.678;
- planimetrical tolerance between points at different levels: 4.000.

3. CONTOURS

The second procedure is devoted to checking the data set, to analyzing clusters vertically and horizontally, to preliminarily distinguishing chains from sowns and to generating relational matching.
The input file could be CCTV.dat or CCTV_rid.dat. In this case, CCTV.dat is used because in the initial data set there are no double points or points out of height. When there are outliers, the file used is CCTV_RID.dat and not the initial data set.
The input data set is listed with the point ID and X, Y and Z coordinates for each points.
The point ID can be the name of the point or the position in the list: both are used and listed.
The output files are:

- CCTV_ord.dat;
- CCTV_intermediate.dat;
- CCTV_assemby.dat.

The service file is CCTV_connections.dat
The data set can be processed in two different ways:

- by checking the input;
- by checking the data set and analyzing it in terms of clusters and relational matching.

The data set is checked in terms of planimetric distance. If points are too close together, only one of them is kept. After this checking procedure, the data set is analyzed.
Input coordinates could be supplied in all possible combinations (X, Y and Z; X, Z and Y; …), the order chosen is 123 (X, Y and Z).
Input data set can also be scaled and rotated, the adopted scales are:

- +1.000 in X axis;
- +1.000 in Y axis;
- +1.000 in Z axis.

and the adopted rotations are:

- 0 in X axis;
- 0 in Y axis;
- 0 in Z axis.

The estimation of the three values needed (the height step between points at consecutive levels, the planimetrical distance between points at the same level and the planimetrical tolerance between points at consecutive levels) is done using "voxel" procedures. The parameters adopted in "contours" procedure are:

- height step: 8.000;

- planimetrical distance: 15.000;
- planimetrical tolerance between points at different levels: 4.000

The first parameter, estimated using the "voxel" procedure, is necessary to do vertical clustering.
For each level, the information is listed in this way:

- ID level;
- number of points belonging to that level;
- level;
- ID point at that level;
- number of vertical clusters.

The extract of the vertical clusters list is the following:

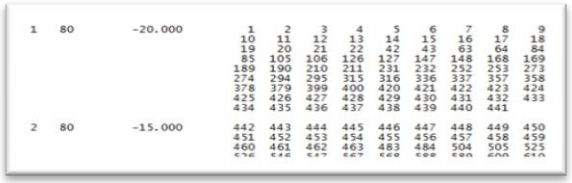


Figure 5. Vertical clusters

In this case, there are 52 vertical clusters so it is provided to the user one 3D plot (in the former image) and 52 2D plots (in the latter two images).

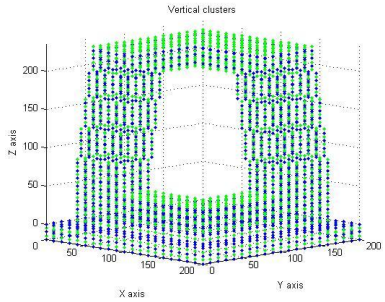
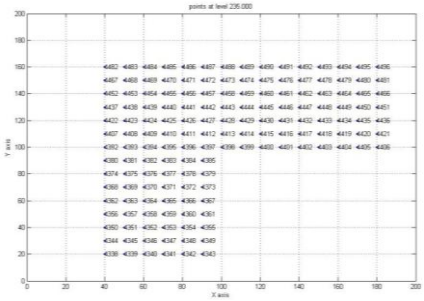


Figure 6. Vertical clusters

Here only two 2D plots are attached:



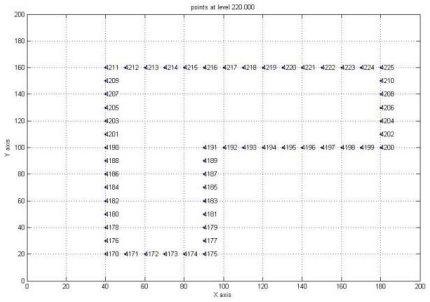


Figure 7. Two of the 52 vertical clusters

The second parameter, estimated using the "voxel" procedure, is necessary to do horizontal clustering. In order to generate contours, finite point sets at each level must be constructed.
These sets could involve all points belonging to a level, or there could be different sets at the same level, if necessary. This second case is displayed well in the picture below.

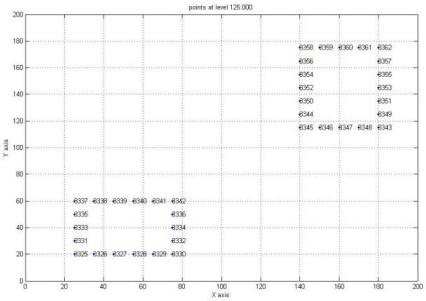


Figure 8. Two horizontal clusters are needed

Each horizontal cluster is described by:

- its horizontal cluster ID level;
- the number of points that belong to the level clustered;
- its level;
- the point ID that belongs to the level clustered;
- the number of points that belong to that level.

The following image shows the case in which there is only one set per level.

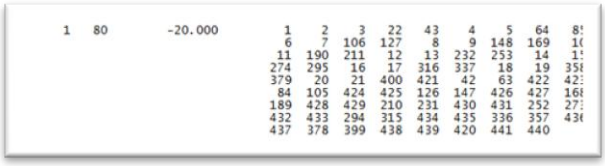


Figure 9. Procedure output with a single horizontal cluster per level

This image shows the case in which two sets are done at the same level.

Figure 10. Procedure output with two horizontal clusters per level

There are 83 horizontal clusters.
The preliminary distinction between sowns and chains is done using this procedure and is made considering firstly the number of points and secondly, and in the case of ambiguity, the local density.
Generally, points that belong to sowns are denser than the ones that belong to chains. If the number of points is the same, it is necessary to check the local density.
The 3D plot is the following:

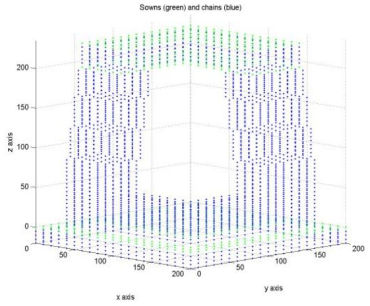


Figure 11. Preliminary distinction between sowns (in green) and chains (in blue)

Although it is a preliminary distinction, all clustered levels are correctly classified, except height 45.000 which is classify as a chain instead of a sown.
The horizontal clusters are classified by:

- horizontal cluster level ID;
- type of clustres;
- height.

At this point, the data set is analyzed in terms of relational matching and the last parameter, estimated with the "voxel" procedure, is used.
As shown in the picture below, if the relational matching is non-correct, the 3D reconstruct is false and the model adopted is non-consisting.

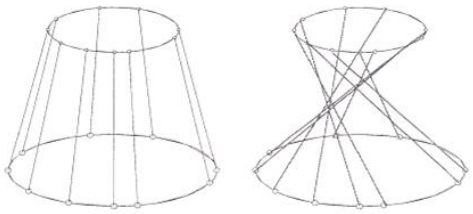
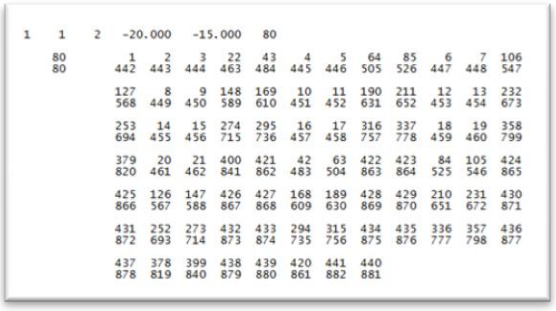


Figure 12. Correct and incorrect matching

The relational matching is done separately for chains and for sowns. The former is classified by:

- relational matching ID (between one chain and another);
- horizontal cluster level ID matched to the other horizontal cluster level ID;
- level matching the other level;
- the number of points belonging to that matching;
- the point ID matched to the other or others point(s) ID;
- the number of sides matched.

The following image shows the output provided:



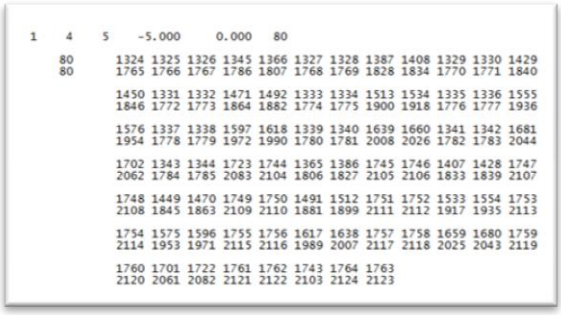
```
1   1   2   -20.000   -15.000   80

   80     1   2   3  22  43   4   5  64  85   6   7 106
   80   442 443 444 463 484 445 446 505 526 447 448 547

       127   8   9 148 169  10  11 190 211  12  13 232
       568 449 450 589 610 451 452 631 652 453 454 673

       253  14  15 274 295  16  17 316 337  18  19 358
       694 455 456 715 736 457 458 757 778 459 460 799

       379  20  21 400 421  42  63 422 423  84 105 424
       820 461 462 841 862 483 504 863 864 525 546 865

       425 126 147 426 427 168 189 428 429 210 231 430
       866 567 588 867 868 609 630 869 870 651 672 871

       431 252 273 432 433 294 315 434 435 336 357 436
       872 693 714 873 874 735 756 875 876 777 798 877

       437 378 399 438 439 420 441 440
       878 819 840 879 880 861 882 881
```

Figure 13. Relational matching for chains

The latter is classified by:

- relational matching ID (between the sown and a chain);
- horizontal cluster level ID matched to the other horizontal cluster level ID;
- level matching the other level;
- the number of points belonging to that matching;
- the point ID matched to the other or others point(s) ID;
- the number of sides matched.

In this specific case, there is a perfect correspondence between each points so the matching type is a one to one and the ID point is related to a single point ID.
The following image shows the output text:



```
1   4   5   -5.000    0.000   80

   80   1324 1325 1326 1345 1366 1327 1328 1387 1408 1329 1330 1429
   80   1765 1766 1767 1786 1807 1768 1769 1828 1834 1770 1771 1840

       1450 1331 1332 1471 1492 1333 1334 1513 1534 1335 1336 1555
       1846 1772 1773 1864 1882 1774 1775 1900 1918 1776 1777 1936

       1576 1337 1338 1597 1618 1339 1340 1639 1660 1341 1342 1681
       1954 1778 1779 1972 1990 1780 1781 2008 2026 1782 1783 2044

       1702 1343 1344 1723 1744 1365 1386 1745 1746 1407 1428 1747
       2062 1784 1785 2083 2104 1806 1827 2105 2106 1833 1839 2107

       1748 1449 1470 1749 1750 1491 1512 1751 1752 1533 1554 1753
       2108 1845 1863 2109 2110 1881 1899 2111 2112 1917 1935 2113

       1754 1575 1596 1755 1756 1617 1638 1757 1758 1659 1680 1759
       2114 1953 1971 2115 2116 1989 2007 2117 2118 2025 2043 2119

       1760 1701 1722 1761 1762 1743 1764 1763
       2120 2061 2082 2121 2122 2103 2124 2123
```

Figure 14. Relational matching for sowns

The total number of sides is also available. In the reconstruction of CCTV, the chain sides matched are 148, the sown sides matched are 18 and the total number is 166.
The specific results are omitted for brevity.

4. CLOSED LOOPS

The third procedure is devoted to building closed loops between all chains and all their projections on sowns.
The input files needed are the ones generated in "contour" procedures and are:

- CCTV_ord.dat;
- CCTV_connections.dat;
- CCTV_assembly.dat.

The output files are:

- CCTV_closedloop.dat;
- CCTV_connections_cla.dat;
- CCTV_contours.dat.

The four values needed are the three generated with the "voxel" procedure and the fourth is a switch parameter in order to choose which type of interpolation should be used.
The switch parameter can assume two values:

- 0, for classic loops;
- 1, for Catmull - Rom loops.

In our opinion, if points represent the vertices of the object, it is better to model the closed loops using the classic way; if the structure is smooth, Catmull – Rom is the best way.
The values of the first three parameters adopted in "closedloops" procedures are:

- height step: 8.000;
- planimetric distance: 15.000;
- planimetric tolerance between points at different levels: 4.000.

For CCTV reconstruction, this parameter is set at 0 because of the shape of the building.
In the first step, the data set is horizontally clustered, in the second, the data is connected one other and each horizontal cluster is finally classify as a chain or as a sown.
This final distinction between sowns and chains is done considering separately points that belong to connections classify as sowns–chains or sowns–sowns and points that belong to connections classify as chains–chains.
For the former the numbers of points is compared: if the first group is composed of a greater number than the other one, the first is classify as a sown while the second one as a chain. On the contrary, if the first group is composed of a lower number than the other one, the first is classify as a chain while the second one as a sown.
For the latter, first the existence of all sowns or all chains is checked. If it is not so, a cycle begins in order to fix all chains making a comparison between the number of points that belong to subsequently horizontal clusters.
At the end, if there are some connections non-classified, the type is set as a sown.
As shown in the following image, all the clusters are correctly classify.

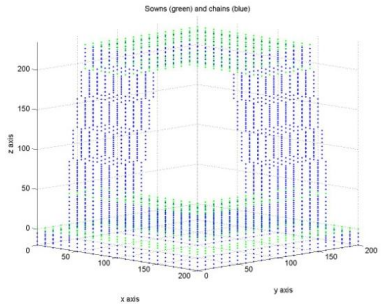


Figure 15. Distinction between sowns (in green) and chains (in blue)

The levels 0.000, 45.000, 205.000 and 235.000 are correctly classify as sowns, all the others as chains.
At this point it is possible to generate the closed loops, one for each horizontal clusters previously done.
All the points must be taken only once and all sides, if taken, only once too.
The generation of closed loops, elementary in elementary cases, becomes complex as the shape of the loop becomes more complicated. In order to force this procedure, the reconstruction of a four letters (F, G, R and W) was preliminarily performed, starting from the rare points.
The next four images show the letters analyzed.

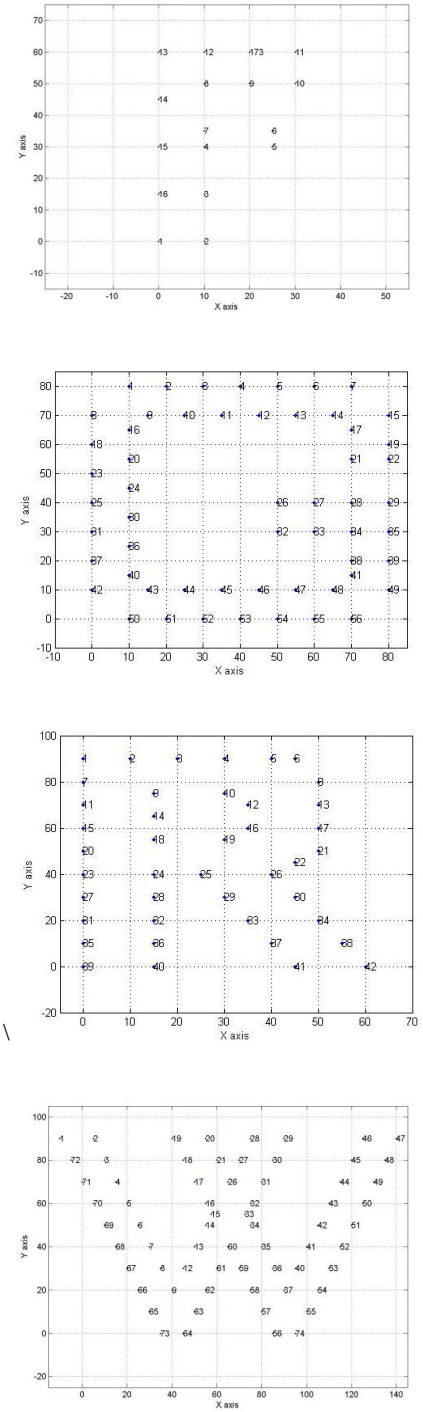


Figure 16. The four letters analyzed

To find a closed loop, first we connect a point to its closest point, next we connect this second point to its closest (excluding the first) and so forth (in order to keep a unique sense of advancement). The sides do not have to cross each other. At this step, a certain number of points will not be connected to the others. Isolated points must be connected in a different hierarchic way:

- with the first point, listed in the previous;
- with the last one, in the same list;
- with all the remaining points;
- with all the ones selected, in the meantime.

Subsequently, isolated loops are connected. Typically these points belong to levels, separated by a unique level, quite often consisting in one or a few points. Moreover further specific analyses are developed, using to the same mechanism.
Only at this step, it is possible to follow the path of the loop. Indeed following a level structure, the path between its root and the farthest leaf identifies the first half of the loop. The second part is done attempting to understand if starting from the farthest leaf the root is reached.
In some cases, it is really impossible to reach the root. Therefore, the whole path is formed by adding single steps, starting from the point where the previous step finishes. At the end, when all points have been taken into account, all the sides not used will be deleted, because lying inside the closed loop they are useless.

4.1. Piecewise Catmull – Rom's lines

It is possible to continuously model every loop with piecewise Catmull – Rom's lines and, starting from 3 points, every straight line passes through the second point and it is parallel to the line joining the first and the third points.
The following image shows the Catmull-Rom closed loop between 5 generic points.

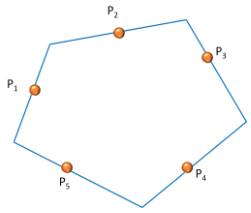


Figure 17. Perimeter with Catmull – Rom

The advantage of a continuous interpolation is the possibility to extrapolate other points, e.g. intersections between consecutive straight lines.

The output given is divided into two groups: the first one for chains and the second for sowns' contours.
The former are organized as follows:

- chain perimeter ID;
- horizontal cluster ID;
- side ID;
- point ID with the ID of the next point on the perimeter;
- the total number of perimetred points that belong to chains.

The latter (closed loops for the sowns' contour) are organized as follows:

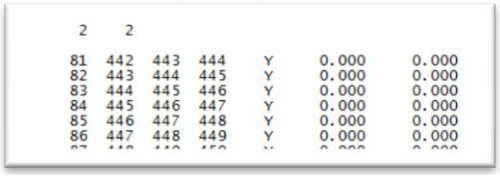
- sowns perimeter ID;
- horizontal cluster (sown) ID and horizontal cluster (chain) ID;
- side ID;
- point ID with other point ID consecutively in the perimetration;
- the total number of perimetred points that belong to chains;
- the total number of perimetred points.

The results of the interpolation in Catmull – Rom perimetretion are divided in contours of chains and contours of sowns. The reports are organized like so:

- interpolation ID;
- horizontal cluster ID;
- side ID;
- ID of the three consecutive points on the perimeter;
- type of straight line
  - $y = ax + b$
  - $x = cy + d$

  in the first case, a y is displayed while in the second, an x;
- the slope of the straight line;
- y-intercept or x-intercept.

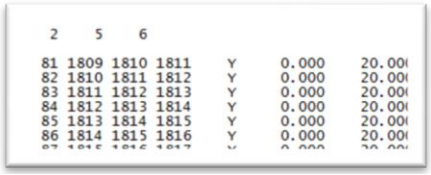
The following image is an extract of the interpolation results for the chains.



2	2					
81	442	443	444	Y	0.000	0.000
82	443	444	445	Y	0.000	0.000
83	444	445	446	Y	0.000	0.000
84	445	446	447	Y	0.000	0.000
85	446	447	448	Y	0.000	0.000
86	447	448	449	Y	0.000	0.000

Figure 18. Chains interpolation

The following image is an extract of the interpolation results for the sowns.



2	5	6				
81	1809	1810	1811	Y	0.000	20.00
82	1810	1811	1812	Y	0.000	20.00
83	1811	1812	1813	Y	0.000	20.00
84	1812	1813	1814	Y	0.000	20.00
85	1813	1814	1815	Y	0.000	20.00
86	1814	1815	1816	Y	0.000	20.00

Figure 19. Contour sowns interpolation

If Catmull – Rom perimetration is set, the straight lines intersect each other so the "closed loop" procedure lists some additional information, divided for chains and sowns:

- ID of the three consecutive points (belonging to chains or to sowns) on the perimetration;
- X, Y and Z coordinates of the intersection point;
- number of intersected points for chains;
- number of intersected points for sowns;
- total number of intersected points.

4.2. Classic way

It is also possible to model every loop in the classic way, in order to make a comparison between the two modes (Catmull-Rom and the classic way).
The following image shows the classic contour between the same five points displayed in figure 17.

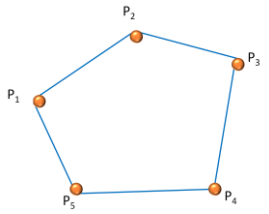


Figure 20. Classic perimeter

Here two images are attached showing two examples of classic perimeters: the former is for chains and the latter is for sowns' contours.

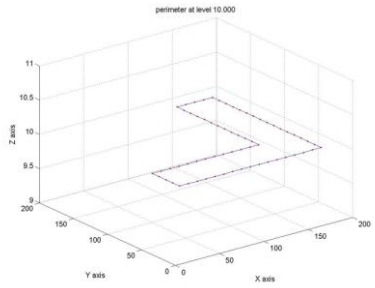


Figure 21. Classic perimeter for a chain (level 10.000)

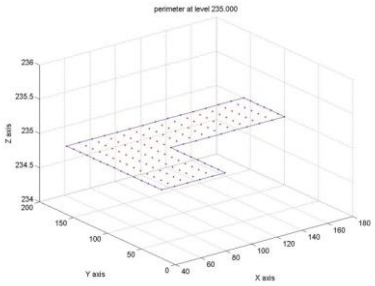


Figure 22. Classic perimeter for the contour of a sown (level 235.000)

The following image shows the perimetretion done in the classic way:

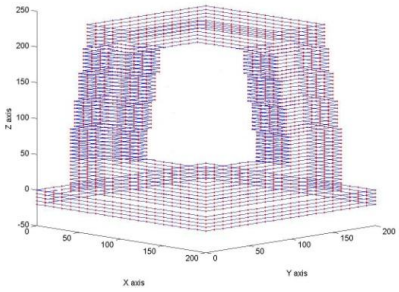


Figure 23. Classic perimeter

5. TRIANGES

The fourth procedure is devoted to connecting points in triangles. These points belong to subsequent chains or subsequent sowns or chains and their projections upon the nearest sowns. There are many possible triangulations, the adopted one is the Delaunay triangulation, where triangles are as equilateral as possible and each triangle must not contain any other points. This last fact implies that a circumcircle passing through three points, should not contain any other point. Delaunay triangulations maximize the smallest of all the angles of the triangles in the triangulation. This triangulation was invented by Boris Delaunay in 1934.
In the Delaunay triangulation of a discrete point set, P in a general position corresponds to the dual graph of the Voronoi tessellation for P.

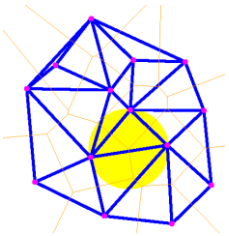


Figure24. Delaunay triangulation (in blue), Voronoi tessellation (in orange) and circumcircle (in yellow).

The input files needed in the "triangles" procedure are:

- CCTV_ord.dat;
- CCTV_connections_cla.dat;
- CCTV_assembly.dat;
- CCTV_closedloops.dat;
- CCTV_contours.dat.

The output files given are:

- CCTV_triangles.dat;
- CCTV_newpoints.dat.

The three values needed are estimated using "voxel" procedures and are:

- height step: 8.000;
- planimetric distance: 15.000;
- planimetric tolerance between points at different levels: 4.000.

The triangulation can connect points belonging to the horizontal cluster, at the same level, as in the extract below:

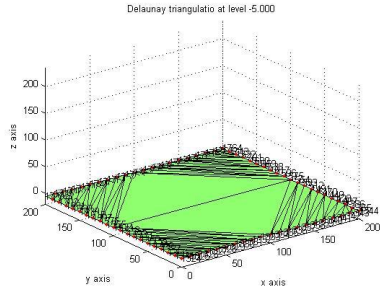


Figure25. Delaunay triangulation (level -5.000)

Each triangulation is associated with a Voronoi tessellation. For lack of space, only one image is attached:

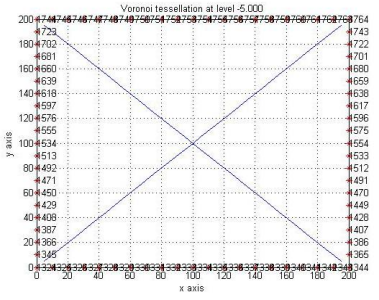


Figure 26. Voronoi tesselletion (level -5.000)

The Delaunay triangulation is also applied to points that belong to a different level. This is done in order to create a rough 3D surface.

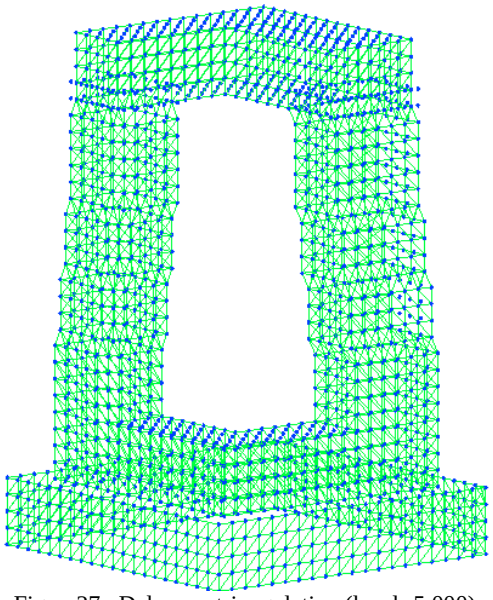


Figure27. Delaunay triangulation (level -5.000)

6. BÉZIER SPLINE

The fifth procedure, still being implemented, is carried out using one of our own softwares called "tri_spline", is devoted to interpolating triangles with triangular splines. For this task, the Bézier spline is used, because it forms continuous surfaces with one continuous derivative surfaces.

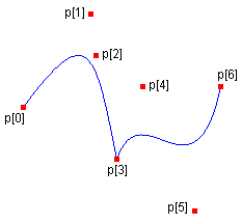


Figure 28. Two splines connected in the middle point

The reconstruction becomes smoother, even if surface breaklines are accepted. Notice that discontinuities of second derivatives are geometrically a bit less evident and therefore neglected.
It is necessary to point out that it is difficult to identify breaklines not highlighted in the input. Another theoretical problem is the analytical complexity of the solution, because the pair of coordinates, which should be used instead of the one dimensional curvilinear coordinate, is not yet globally defined.
At the moment, some theoretical aspects should be studied in detail, recognizing the elegance of the Bézier spline, even in one-dimensional space.

7. FUTURE WORKS

The estimates of the first three parameters done using the "voxel" procedures are an underestimation of the real values.
The second problem to be analyzed is the possibility to model almost-horizontal sowns correctly. In this specific case, the top of CCTV is not horizontal, while in the 3D model it is analyzed and treated as a horizontal plan.
The third problem is the fact that the Bézier spline is well known, but there is no definite coordinate pair so some extra studies are needed in order to find a solution.

8. COMMENTS

Our first comment regards the results as a whole: they try to take a step forward compared to a wide range of previously collected samples. In any case, all programs (except for computer graphics applications) are written, implemented, tested and used forming a library of our own softwares.
The second comment relates to the approach used: the user is provided a graphic result and *.txt files where all the results are analyzed.

9. REFERENCES

George A. Liu Jwh, 1981. *Computer solution of large, sparse, positive and definite system*, Prentice all, Engliwood Clifts, New Jersey.

Hawkins D.M., 1980. *Identification of outliers*, Chapman and Hall, London.

Jodidio P., *Architecture now!*, volume 7, 2010.Taschen.

Kaufman L. Rousseeuw Pj, 1990. *Finding groups in data*, Prentice all, Engliwood Clifts, Wiley&Sons, New York.

Morteson Me, 1997. *Geometric modeling*, Wiley&Sons, New York.

Naldi G., Pareschi L., 2007. *Matlab, concetti e progetti*. Apogeo, Lavis (TN).

Preparata F. P., Shamos M. I., 1985. *Computational Geometry, an Introduction*, Springer, New York.

Thurston W., 1997. *Three-dimensional Geometry and Topology*, Princeton University Press, Princeton.

Week J., 1985. *The shape of space*, Dekker, New York.