POLITECNICO DI TORINO
Scuola di Dottorato

POLITECNICO DI BARI

POLITECNICO DI MILANO

SCUOLA INTERPOLTECNICA DI DOTTORATO (SIPD)
DOCTORAL PROGRAM IN ENVIRONMENTAL AND TERRITORIAL SAFETY AND CONTROL

3D MODELING
STRUCTURES, INFRASTRUCTURES, AQUIFERS AND SHRUB

Doctoral dissertation name:
Valentina Forcella

Tutor:
Prof. Luigi Mussio

The chair of the doctoral program:
Prof. Fernando Sansò

2012 – 25° cycle

To my dad

ABSTRACT

This dissertation contributes to the 3D modeling field. In fact, the goal of this work is to obtain 3D models automatically from a discrete data set.

Each 3D model is obtained using different softwares packages and methods. The former are Fortran, Matlab, ArcGIS, Model Builder, Arcpy, and Visual Nature Studio. The latter are outlier detection, voxels, clusters analysis, feature extraction, relational matching, Delaunay triangulation, and Bruckner profiling. The order in which these methods are used depends on the data types. The input point cloud can represent a structure, an airport, an aquifer or a forest.

The study areas are five:

- the China Central Television Tower Headquarters in Beijing, China
- an airport in Italy
- the aquifer in Milan, in Italy
- Levy County in Florida, U.S.A.
- Marion County in Florida, U.S.A.

Some problems must be solved to obtain a consistent 3D model. The first problem originates from to the fact that the input data are discrete, not equally spaced or uniform and the outputted model must be continuous.

Secondly, the input information is geometric, not topological. To obtain a 3D model, topological information must be collected from the input data. Finally, if the input data contains some outliers, the predicted values will be wrong. To solve this problem, suitable procedures must be set up.

Although there are other important issues, I would like to point out that this dissertation is not concerned with the different techniques available to obtain the data and to check the completeness of the data. Furthermore, this dissertation is not concerned with the rendering techniques, added light source properties, or created realistic and stylish looking models. The 3D models are obtained using Matlab ™.

SUMMARY

I decided to divide the summary of my dissertation into four sections:

- objective of the dissertation,
- methodology,
- problems and proposed solution, and
- conclusion.

1. OBJECTIVE OF THE DISSERTATION

This dissertation contributes to the 3D modeling field. In fact, the goal of this work is to obtain 3D models automatically from a discrete data set.
Some of the ideas behind this topic is that obtaining 3D models is time consuming and difficult, it requires softwares and skills and the methodology is different depending on the data type, the operator and the user.
I decided to find a general method to obtain the 3D model using different kind of point cloud. For this reason, the study cases are five and are the following:

- the China Central Television Tower Headquarters in Beijing, China
- the Orio al Serio International airport in Bergamo, in Italy
- the aquifer in Milan, in Italy
- Levy County in Florida, U.S.A.
- Marion County in Florida, U.S.A.

The data set is composed of the coordinates of the points and extra information might be included in the input data set. As mentioned, the input point cloud can represent a structure, an airport, an aquifer or a forest. The objective of the research is therefore to develop a sequence of procedures in order to obtain the 3D model automatically, independently to the type of point cloud in input.

To do so, some softwares packages and methods have been taken into consideration. The following sections and the dissertation provides a detailed description of each step.

2. METHODOLOGY

Each 3D model is obtained using different softwares packages and methods. The former are Fortran, Matlab, ArcGIS, Model Builder, Arcpy, and Visual Nature Studio. The latter are outlier detection, voxels, clusters analysis, feature extraction, relational matching, Delaunay triangulation, and Bruckner profiling.

In the following, a brief introduction to each software is presented.

Fortran was originally developed by IBM in 1957 for scientific and engineering applications. It is designed to allow easy translation of math formulas into code. Some eldest steps of my procedures are coded using Fortran.

MATLAB was started developing in the late 1970s by Cleve Moler, the chairman of the computer-science department at the University of New Mexico. The name MATLAB comes from MATrix LABoratory. Matrix because the data type used is the matrix, laboratory because MATLAB is used at school for didactics and research. The newest procedures, 3D models, 2D plots, and 3D plots are obtained using Matlab.

Esri's ArcGIS is a geographic information system (GIS) for working with maps and geographic information. It is used for creating and using maps, compiling geographic data, analyzing mapped information, sharing and discovering geographic information, using maps and geographic information in a range of applications, and managing geographic information in a database [Wikipedia].

Python was introduced to the ArcGIS community at 9.0. Python is a free, cross-platform, open-source programming language. ArcGIS 10 introduces ArcPy. It develops Python scripts while offering code completion and integrated documentation for each function, module, and class. In this way, only the toolbox is needed. Model builder is a way to automate a series of tools. It is a part of the ArcGIS geoprocessing framework.

Visual Nature Studio 3 makes creating complex scenes and faster rendering. Visual Nature Studio (VNS) includes foresters, engineers, landscape architects, GIS professionals, historians, graphic artists and many, many others. It is possible to import the GIS data and make a photorealistic image. The last softwares are used when the input is LiDAR data.

In the following, a brief introduction to each method and technique is presented. They are several and come from different fields. Outlier detection is a primary step towards obtaining a coherent model. Different point IDs could have the same coordinates or different points with different coordinates could have the same point ID. Some points could be out of height. The aim is to decide which points must be deleted and which one must be used.

A second method is the subdivision of the data into volumetric element. A voxel (volumetric picture element) is an element of the total volume, representing a value on a regular grid in three dimensional space [Wikipedia]. The 3D space is split by equal intervals along the three orthogonal directions to produce a voxel representation. The zero order of voxel contains 1 cube, the first order of voxel contains 8 cubes, the second one contains 64 cubes, the third one 512 cubes, the fourth one 4096 cubes, the fifth one 4096 cubes, the sixth one 32768 cubes and so on.

A third technique makes use of the subdivision of the data into groups. The cluster analysis is the process of organizing objects into groups whose members are similar in some way [Anderberg, 1973, Dubes and Jain, 1988] and are dissimilar to objects belonging to other clusters. Clustering algorithms may be classified in different ways.

A fourth method is the extraction of features from the point cloud. One of the possible feature is the perimeter. In fact, all the points that belong to each clusters must be perimetred with a closed loop in order to obtain a continuous 3D model. The generation of closed loops is easy only in elementary cases. It becomes more complex as the shape of the loop becomes more complicated. There are two different types of closed loops. The former is the piecewise Catmull – Rom's lines and the latter one is the classic one.

A fifth step is the relational matching. The relational matching method is used to search the correct correspondences between points that belong to consecutive layers to generate a vertical boundary.

A sixth technique is the Delaunay triangulation. It is possible to connect a set of points with triangles and triangulation is the name of this method. There are many possible triangulations, the adopted one in this dissertation is the Delaunay triangulation.

A seventh method is the interpolation, starting from the input data. The aim is to obtain a denser information. It is possible to do this using different techniques. The first one is the parametric equation of a line passing through two points. It is used to obtain the coordinates of new points. The lines are the sides obtained by Delaunay triangulation and the points are the ones of the point cloud. It is also possible to obtain the coordinates using the mean or the weight mean of the point cloud.

The Bruckner diagram analysis, highly adopted in highway engineering, consists in the calculation of cut and fill volumes of soil during a road construction project. In particular, the Bruckner diagram can inform engineers about soil handling quantities to be provided or removed for each road section. Therefore, analyzing soil quantities among the whole longitudinal road profile allows the optimization of materials to be handled in order to minimize costs and hauling distances. The Bruckner diagram can be obtained by integrating the areas diagram for two consecutive road sections.

The results of the study indicate that all four types of input point cloud can be processed using the mentioned softwares package and methods. The only thinks that depends on the data type is the order in which these methods are used.

3. PROBLEMS AND PROPOSED SOLUTIONS

Some problems must be solved to obtain a consistent 3D model. The first problem originates from to the fact that the input data are discrete, not equally spaced or uniform and the outputted model must be continuous. To solve this problem, interpolating methods are used.

Secondly, the input information is geometric, not topological. To obtain a 3D model, topological information must be collected from the input data. If the shape of the model is non-stellar, to reach all the points, it is necessary to go outside the structure. If the shape of the point cloud is multi-connected, it means there is a hole. The 3D model must have the same hole. In addition, if the input point cloud has some holes, it is not possible to directly apply Delaunay triangulation and some adaptations must be done.

Thirdly, if the input data contains some outliers, the predicted values will be wrong. To solve this problem, suitable procedures must be set up using outlier detection tools.

Fourthly, if the input point cloud represents a concave body, Delaunay triangulation does not work so changes have to be found and applied. If the input point cloud represents a convex body, Delaunay triangulation perfectly works so no adaptations have to be done.

As mentioned in the previous section, the methods and the order they are used are the key to obtain the 3D model automatically.

As stated earlier, five different input point cloud and four different data type are taken into account. In the following, a brief scheme to the methodology adopted is described.

The first case study is a structure and is the China Central Television Tower Headquarters in Beijing, China.

I chose this building because its shape is not common and my master's thesis case study was this one. The shape of this building is non-stellar, concave and multi–connected.

The integrated techniques and the procedures used are the following:

- the first one is used to check the input data set,
- the second one is the "voxel" procedure that is a method based on octree encoding,
- the third one is called is devoted to doing the cluster analysis,
- the fourth one is used to extract features from the data set,
- the fifth one is devoted to doing the relational matching, and
- the sixth one is called "triangle" procedure and is devoted to connecting points with triangles.

The second case study is the Orio al Serio International airport, in Bergamo, Italy.

The purpose is to obtain a 3D model automatically. In addition, the area of each cross-section is required. The areas must be gotten automatically.

The applications are numerous. First of all, it is possible to re-applied this workflow to any airport independently to the Aerodrome Reference Code. Secondly, the automation of the process makes the goal of the 3D model easier and fast. Thirdly, it is possible to compare the input data with a different type of airport if the purpose is to get a new airport with a different Aerodrome Reference Code compared to the starting one.

The shape of this building is stellar, convex and single–connected.

The optimized 3D model is generated using some integrated techniques and the following procedures:

- the first one is used to check the data set and to provide the first 3D plots,
- the second procedure is devoted to generating the 3D model,
- the third one is used to analyze the input point cloud and the 3D model in terms of clusters,
- the fourth procedure is called "relational matching" and is devoted to connecting points belonging to different clusters,
- the fifth one is called "triangle" procedure and is devoted to connecting points with a Delaunay triangulation,

- the sixth one is called “Bruckner” procedure and is used to evaluate the volume of soil to be handled, and
- the seventh procedure is devoted to obtaining the optimized 3D model.

The third case study is the aquifer of the city of Milan (Lombardy, north of Italy) and a nearby area of about 580 km^2.
The purpose is to obtain a different workflow in case a deterministic interpolation is required.
The shape of this building is stellar, convex and single–connected.
It is possible to solve the highlighted problems subdividing the data processing into six steps.

- First of all, the input data are checked in order to detect and remove the outliers.
- Secondly, the information is divided into clusters. This procedure is carried on to assign observations into subset.
- Thirdly, each cluster is interpolated using the Delaunay triangulation. This method interpolates scattered points and a continuous field is obtained.
- Fourthly, closed paths between points that belong to a specific layer are built. It is necessary to bound each layer in order to know where to interpolate and where to extrapolate.
- Fifthly, it is necessary to search the correct correspondences between points that belong to consecutive layers to generate the vertical boundary.
- Sixthly, it is necessary to obtain new information starting from the Delaunay triangulation. This procedure is carried on using the interpolation method explained in chapter 1, section 7.

The fourth and fifth case study are Levy County and Marion County. Staring from a LiDAR point cloud, the purposes of this chapter are two: the former is to obtain a better classification of the input data and the latter is to obtain density maps.
All the input points are classified into three classes: class 1 includes unclassified points, comprising vegetation, buildings, noise, etc; class 2 includes ground points; and class 9 includes water points. It is possible to better classify class 1 into new sets: roads and vegetation. This last set contains a sub-set called shrub. A shrub or bush is a woody plant characterized by several stems arising from the base and height usually under 19.658 ft (6 m) tall.
Once the shrub is detected, the idea is to cluster the shrub into subsets according to the location and the density. The shrub can be divided into three type of density: light, medium and heavy. Density maps can be used by rescuers during rescue emergency. This work can be used as the input of fire disaster pre-plan. Indeed mathematical fire behavior models require information on the location, the density and the species of each tree. Using LiDAR data, it is possible to detect the location, the height and the density, but not the species.
In order to sub-classify class 1, these different tools have to be used:

- LAS to multipoints: this tool imports all the files in LAS format into a new multipoint feature class. The LAS to multipoints step is applied twice: the first time to obtain the non-ground points using the Model Builder "LAStoMPng". The second time to obtain the ground points using the Model Builder "LAStoMPg".

- Point to raster: the point to raster converts point features to a raster dataset. The point to raster step is applied twice: the first time to obtain the Digital Surface Model (DSM), the second time to obtain the Digital Elevation Model (DEM).
- Raster to point: the raster to point conversion tool converts a raster dataset to point features. The Raster to Point Conversion tool converts a raster dataset to point features. For each cell of the input raster dataset, a point will be created in the output feature class.
- Select by attributes: it adds, updates, or removes a selection on a layer or table view based on an attribute query. To select all the points within a certain height, Select Layer By Attribute (Data Management) is used. This step is applied four times:

    - vegetation high lower than 0 ft
    - Height between 0 ft and 3.28 ft (1 m)
    - Height between 3.28 ft (1 m) and 6.56 ft (2 m)
    - Height between 6.56 ft (2 m) and 19.658 ft (6 m)

- Split: splitting the input features, it creates a subset of multiple output feature classes.
- Feature to polygon: it creats a feature class containing polygons generated from areas enclosed by input line or polygon features.
- Clip: it extracts input features that overlay the clip features. This tool cuts out a piece of the roads feature class using the footprint feature in the new feature class called clipped roads.
- Selects features in a layer based on a spatial relationship to features in another layer.
- Point density: it calculates a magnitude per unit area from point features that fall within a neighborhood around each cell.

4. CONCLUSIONS

To conclude, it is possible to obtain the 3D models automatically starting from different types of point clouds. Not only convex bodies, but also concave, non-stellar and multi-connected can be processed. The explanation and the implications to obtain the 3D model will be explained in the introduction and in the other chapters if the input point cloud represents a concave, non-stellar, and multi-connected.
I made two assumptions:

- the input data is complete
- the input data is obtained with different techniques, to process the data a file *.txt is required with the coordinates of the points and their IDs.

The geometric characteristics of the model are put first than the graphic output. It implies I focused my attention on how to obtain the 3D models automatically, not on the graphical restitution. I used Matlab to obtain the 3D models, fully aware there are softwares whose purpose is to renders 3D models. The models do not contain shadows, light effects, and so on.
There are some limitations. First of all, it is necessary to find out different parameters. The number of the required parameters was kept as low as possible. Different parameters are needed depending

on the different types of input point cloud. For this reason, different codes were written to obtain the different values automatically.

It is possible to obtain the 3D model of an input point cloud if it represents a structure, an infrastructure, an aquifer or a shrub.

It is possible to obtain the 3D model automatically using Matlab and ArcPy and scripts wrote for the purpose. It is necessary to take into account also some limitations. For example, some parameters depend on the specific input point cloud and not on the type of input point cloud.

The dissertation is divided into three parts. The first part describes contains an explanation of each technology and some modifications or alterations done in order to better apply it to the case studies. It also contains some open problems and possible solutions. Within the second section a detailed description of each study case is included. As mentioned, the study cases are:

- the China Central Television Tower Headquarters (CCTV) in Beijing, China
- an International airport in the North of Italy
- the aquifer in Milan, Italy
- Levy County in Florida, U.S.A.
- Marion County in Florida, U.S.A.

The third part contains two appendixes. The former describes an explanation of each software and the latter one describes the application in 3D of the Bézier splines. Bézier splines are defined and used in 2D space using a one coordinates. The definition of the Bézier splines in a 3D space is not available because the two coordinates that define the Bézier spline is still an open problem.

I am fully aware that I am inside a path. Some people before me studied the 3D modeling field and so will other do in the future. It is not possible to isolate works so this dissertation should be located inside a bigger workflow, made by past studies and future researches.

Although there are other important issues, I would like to point out that this dissertation is not concerned with the different techniques available to obtain the data and to check the completeness of the data. Furthermore, this dissertation is not concerned with the rendering techniques, added light source properties, or created realistic and stylish looking models.

PART I

CHAPTER 1

Introduction

In 3D computer graphics, 3D modeling is the process of developing a mathematical representation of any three-dimensional surface of object via specialized software. The product is called a 3D model. The model can also be physically created using 3D printing devices.

Models may be created automatically or manually. The manual modeling process of preparing geometric data for 3D computer graphics is similar to plastic arts such as sculpting. 3D models represent a 3D object using a collection of points in 3D space, connected by various geometric entities such as triangles, lines, curved surfaces, etc. Being a collection of data, 3D models can be created by hand, algorithmically, or scanned.

3D photorealistic effects are often achieved without wireframe modeling and are sometimes indistinguishable in the final form. 3D printing is a form of additive manufacturing technology where a three dimensional object is created by laying down successive layers of material. [Wikipedia]

This dissertation tries to find to answer the following questions: how is it possible to get 3D models automatically using one or few not expensive softwares? How is it possible to combine different techniques to obtain 3D model if the input point cloud represents a structure, an infrastructure, an aquifer or low vegetation?

The 3D modeling field can be split in different areas. They are the following: 3D outdoor modeling, 3D indoor modeling, 3D printing, and augmented reality. To get the goal using different techniques and softwares can be used.

It is possible to obtain 3D modeling using an application called Rhino 3D. [Gabriel Mathews].

Photosculpt Textures Software V2 will create a high definition 3D model of the subject automatically starting from two photographs. Then you can render with any 3D render.

The program generates the model within minutes simply taking two pictures of the subject. It is possible to display the model using 3D meshes. It does a great job on 3D textures and a great job on natural surfaces. [Copyright hipe0 2009].

A team headed by Professor Avideh Zakhor developed a laser backpack that can scan the surrounding and then it creates instantly 3D model. The goal is to obtain fast, automated, 3D model of building interiors without the human intervention.

It combines miniature lasers with an inertial unity, and cameras capture a panorama. The idea is to wear a backpack, to walk inside the building, when it is done a button is pushed and out comes the model. New breed miniature lasers with an inertial management like the one that guide missiles are used. The IMU localizes the backpack, lasers generates the geometry and four cameras generate the texture map. All three are then fused for precise navigation.

The 3D outdoor modeling combines airborne and ground-based laser scanners and cameras images.

A truck equipped with one camera and two fast, inexpensive 2D laser scanners, being driven on city streets under normal traffic conditions. The Ground-based modeling and airborne modeling are merged, registered and fused in the 3D city model. [Christian Fruh and Avideh Zakhor]

A printer turns 3D models into real physical things. 3D printing will soon allow digital object storage and transportation, as well as personal manufacturing and very high levels of product customization. An example of a 3D printer is Ponoko the Makerbot. [Gabriel Mathews].

If the 3D model represents an aquifer, the 3D models of the aquifers are generated using interpretative stratigraphic sections or, more recently, the kriging technique. This last method is a stochastic one and the aim of this procedure is to estimate the value of the field at an unobserved location from a combination of the observations [Paola Gattinoni].

If the data of the input point cloud are LiDAR data, a combination of analog photointerpretation, digital softcopy photogrammetry, geographic information system (G.I.S.) and global positioning system (G.P.S.)- assisted field data collection procedures were employed in the construction of the vegetation data base. [M. Madden, Welch, T. Jordan et al.].

If the goal is to obtain a 3D model of an airport, the tools used are AutoCAD and cross-sections. Each section is joint to the previous and to the following manually. [Filippo Giustozzi].

For all these reasons, this dissertation tries to find an almost unique workflow to obtain automatically the 3D model. The goal is also to be able to repeat the modeling on the same type of input point cloud but, for example, in a different location.

The previous solutions adopted have some limitations and some problems must be solved.

For example, Photosculpt Textures Software V2 is only useful for textures with bump maps, not entire buildings or mountains. It does not work 360 degree perfectly.

Another problem is the following: many people may do not want or be able to afford a 3D printer. Prices for most commercial/industrial 3D printers tend to start in the ten-to-twenty thousand dollar bracket and spiral upwards.

Using a stochastic method to interpolate data, the value estimated in a point where the measure was acquired does not coincide with the measure itself.

Achieving the 3D model of the airport through the manually cross sections is time consuming and the repeatability is missing.

So far no one has investigated the link between different methods and few softwares to obtain 3D models automatically. And even using well-known technology, such as Delaunay triangulation, some problems must need to be solved. In fact, the above-mentioned solution do not apply to a concave body case.

Also the Bruckner profiling is a well-know methodology, but it is not possible to apply it to an airport without drawing cross section manually.

To conclude, the purposes are two: the first one is the outputted automatic 3D model and the repeatability of the process.

I divided my dissertation into three parts. The first one contains an explanation of each technology and some modifications or alterations done in order to better apply it to the case studies. It also contains some open problems and possible solutions.

Within the second section a detailed description of each study case is included. As mentioned in the summary, the study cases are:

- the China Central Television Tower Headquarters (CCTV) in Beijing, China
- an International airport in the North of Italy
- the aquifer in Milan, Italy
- Levy County in Florida, U.S.A.
- Marion County in Florida, U.S.A.

The fist one is used to obtain a proper workflow for a structure. The CCTV was chosen because of its unusual shape.

The second study case was chosen because a previous study on that airport was done in the Infrastructure department at Politecnico di Milano, in Italy.

The third study case is the aquifer in Milan and a nearby area of about 580 km^2. A previous model was done by Gattinoni.

The fourth and fifth case studies are Levy County and Marion County in Florida, U.S.A. I spent six months at the Center for Geospatial Research (CGR), formerly the Center for Remote Sensing and Mapping Science (CRMS). Doctor Marguerite Madden is the Director of the Center for Geospatial Research and Doctor Jordan, Thomas R. is the Associate Director of the Center for Geospatial Research.

The third part contains two appendixes. The former describes an explanation of each software and the latter one describes the application in 3D of the Bézier splines. Bézier splines are defined and used in 2D space using a one coordinates. The definition of the Bézier splines in a 3D space is not available because the two coordinates that define the Bézier spline is still an open problem.

CHAPTER 2

Techniques

This chapter explains the different techniques used to obtain 3D models automatically. They are several techniques and they come from different fields.
The techniques used and their order depend on the input data type. In fact, the input point cloud can represent a structure, an airport, an aquifer or shrubs.
The following list provides the name of the techniques utilized:

- outlier detections,
- voxel distribution,
- cluster analysis,
- relational matching,
- features extraction,
- Delaunay triangulation,
- interpolation,
- Brückner profiles, and
- stabilized Erone.

For each method at least one applied example is provided. The proposed examples are a structure, an airport and an aquifer.
The first one is a not an existing building, but it resembles a derrick and a frame structure. The shape of this case study is complex because it is concave, non-stellar, and multi-connected. This building is used to test multiple software programs I wrote using Matlab and Fortran. Figure 2.1 shows the point cloud that represents the building.

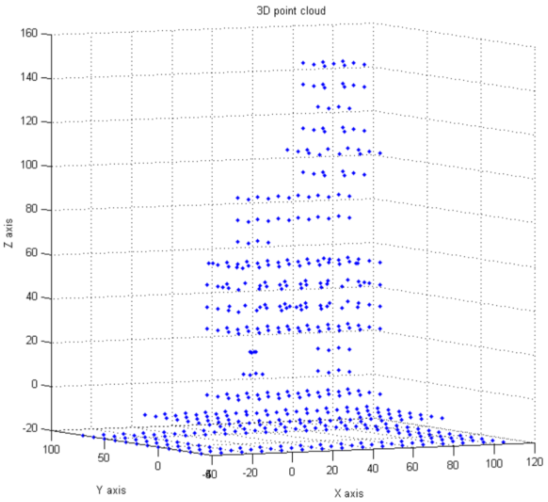


Figure 2.1 Structure: input point cloud

The second one is an airport and is also analyzed in chapter 7. An airport is composed of several parts. They are the Clear Way, RESA, center line, runway, runway shoulders, Cleared and Grated Area (CGA) and runway strip.
Figure 2.2 shows the point cloud that represents the airport.

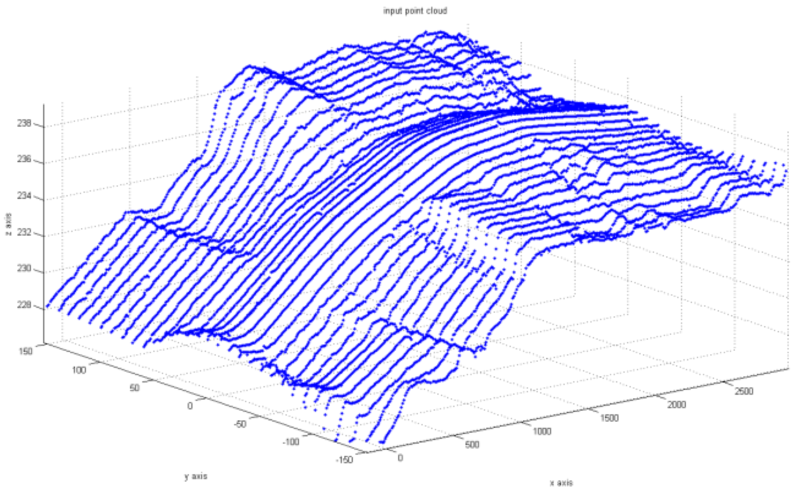


Figure 2.2 Airport: input point cloud

The third study case is the aquifer, which is also used for the study case in chapter 8. The aquifer is modeled by points defined by three coordinates and the type of layer to which they belong. There are four different layers:

- land surface layer,
- Aquitard up layer,
- Aquitard down layer, and
- Aquiclude layer.

Figure 2.3 shows the point cloud that represents the aquifer.

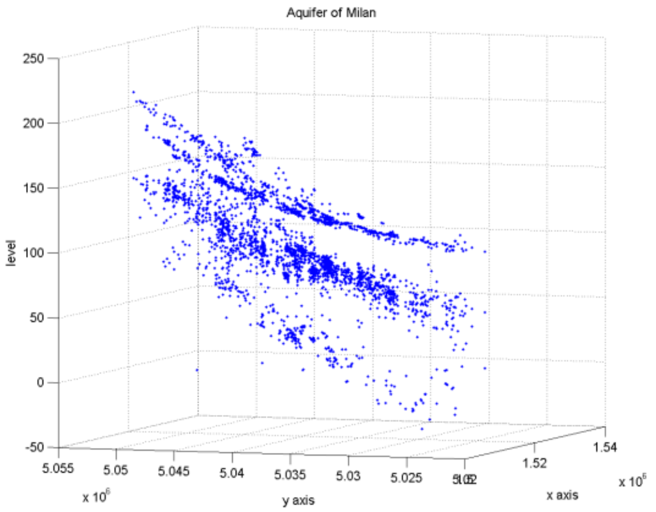


Figure 2.3 Aquifer: input point cloud

Each method is explained in detail in sections 1.1 to 1.9.

2.1. OUTLIER DETECTION

Outliers are values that are numerically distant from the rest of the data and must be recognized and might be excluded from further data analysis [V. Pupovac and M. Petrovečki]. The presence of outliers can dramatically change the shape of 3D model. For this reason, it is necessary to remove the outliers from the input point cloud.

Detecting and correcting outliers automatically is impossible because there are many kinds of outliers and their presence is different in each input point cloud. Nevertheless, I found three kinds of outliers in the input point clouds I analyzed.

First of all, different point IDs could have the same three coordinates. Second, different points with different coordinates could have the same point ID. Lastly, some points could be anomalies.

The aim is to decide which points must be deleted and which one must be processed, independently from the type of outliers.

A short example will clarify the adopted procedure: there are five points in figure 2.4. The points P_2 and P_5 have different ID, but the same coordinates.

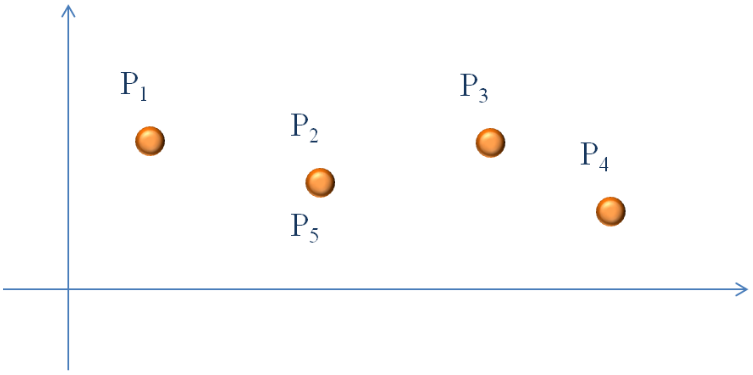


Figure 2.4 First type of outliers

Only one of the two points is processed, as shown in the following picture.

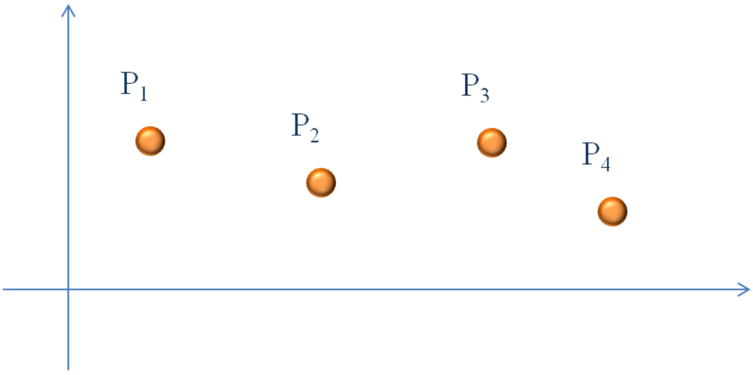


Figure 2.5 Points after outlier detection

The second kind of outlier is the following: different points with different coordinates have the same point ID. In this case, the point ID of one of these points is changed.
In figure 2.6 the third kind of outlier is shown. There are seven points and the point P_7 is classified as belonging to the same point cloud. That point is an outlier.

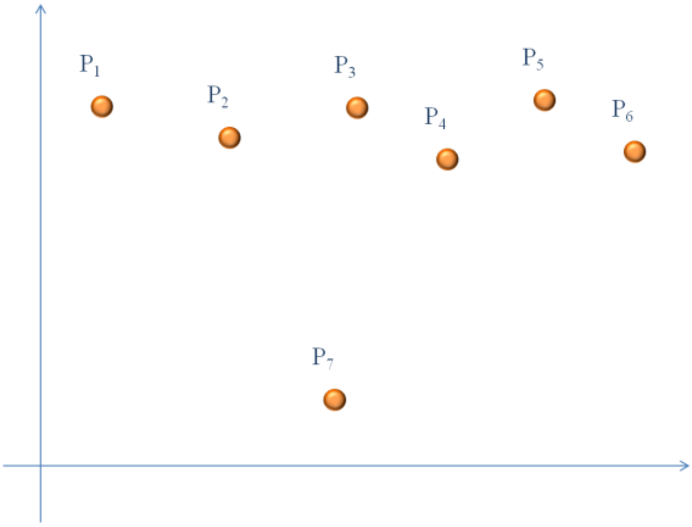


Figure 2.6 Points before the outlier detection

2.2. VOXELS

A voxel (volumetric picture element) is an element of the total volume, representing a value on a regular grid in three dimensional space [Wikipedia].
As a pixel represents the information about a 2D field, a voxel is the equivalent in the 3D world. Voxels are used for analysis and visualization of 3D-dimensional distribution of data.
The 3D space is split by equal intervals along the main orthogonal to produce a voxel representation.
Figure 2.7 shows an example of a voxel subdivision.

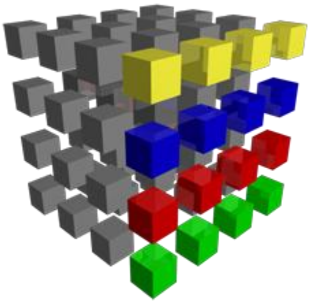


Figure 2.7 Voxel

The number of voxels used in a calculation can vary.

The optimal data allocation is reached through division of the whole data set into data subsets and keeping only non-zero pieces of data in memory.

It is possible to start the subdivision into voxels using the maximum dimensions of the scene to adjust the dimensions of the starting cube. Then the starting cube is subdivided into 8 volumes. Each new volume is subdivided into 8 volumes if the starting volume is not empty. This procedure goes on until the volumes are empty.

The procedure described in the three steps imply the fact that the zero order of voxel contains 1 cube, the first order of voxel contains 8 cubes, the second one contains 64 cubes, the third one 512 cubes, the fourth one 4096 cubes, the fifth one 4096 cubes, the sixth one 32768 cubes and so on.

The following pictures show the starting volume, and then subdivided into different order of voxels.

Figure 2.8 shows the sub-division of the input point cloud into the zero order of voxel. There is only 1 cube.

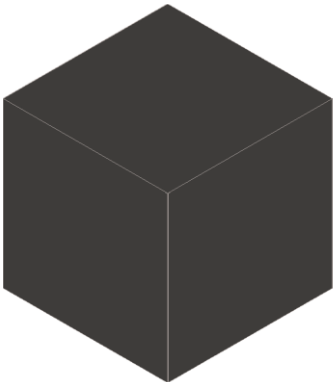


Figure 2.8 Voxel, 0° order

In this case the amplitude in x is equal to xmin – xmax, the amplitude in y is equal to ymin – ymax, and the amplitude in z is equal to zmin – zmax.

Figure 2.9 shows the sub-division of the input point cloud into the first order of voxel. There are 8 cubes.

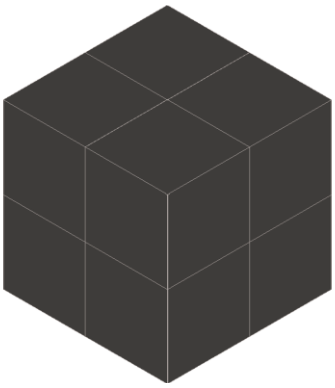


Figure 2.9 Voxel, 1st order

In this case the amplitude in x is equal to xmin – xmax divided by 2, the amplitude in y is equal to ymin – ymax divided by 2, and the amplitude in z is equal to zmin – zmax divided by 2.
Figure 2.10 shows the sub-division of the input point cloud into the second order of voxel. There are 64 cubes.

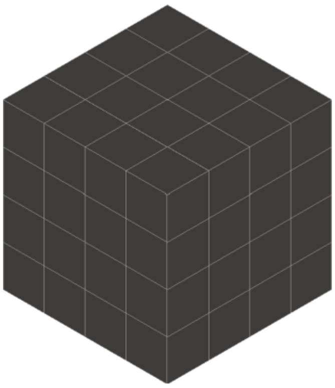


Figure 2.10 Voxel, 2nd order

In this case the amplitude in x is equal to xmin – xmax divided by 4, the amplitude in y is equal to ymin – ymax divided by 4, and the amplitude in z is equal to zmin – zmax divided by 4.
Figure 2.11 shows the sub-division of the input point cloud into the first order of voxel. There are 512 cubes.

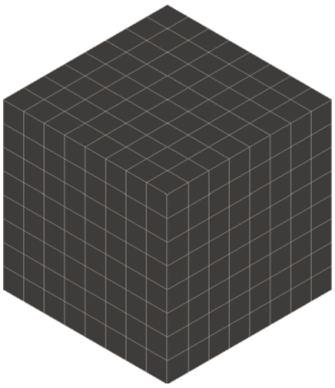


Figure 2.11 Voxel, 3rd order

In this case the amplitude in x is equal to xmin – xmax divided by 8, the amplitude in y is equal to ymin – ymax divided by 8, and the amplitude in z is equal to zmin – zmax divided by 8.

Figure 2.12 shows the sub-division of the input point cloud into the first order of voxel. There are 4096 cubes.

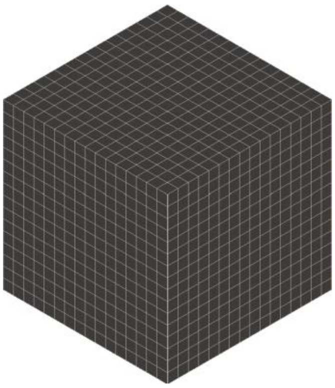


Figure 2.12 Voxel, 4th order

In this case the amplitude in x is equal to xmin – xmax divided by 32, the amplitude in y is equal to ymin – ymax divided by 32, and the amplitude in z is equal to zmin – zmax divided by 32.

Most literature seems to point to an Octree as the most suitable structure for a hierarchical structure of a voxel grid. There are two kinds of algorithms for ray-octree intersection: bottom-up, which works from the leaves back upwards, and top-down, which is basically a depth-first search.

This enables spatial partitioning, down sampling and search operations on the point data set. Each octree node has either eight children or no children. The root node describes a cubic bounding box which encapsulates all points. This representation is shown in figure 2.13.

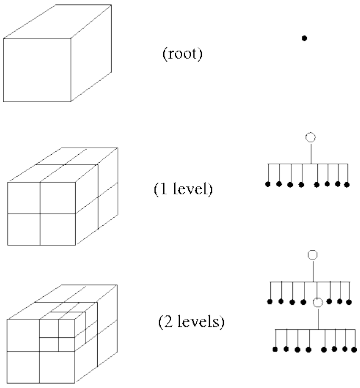


Figure 2.13 Octrees

A generalization of voxels is doxels, or dynamic voxel. This is used in the case of a 4D dataset, for example, an image sequence that represents 3D space together with another dimension such as time. It allows the representation and analysis of space-time systems.

There are efficient methods for creating a hierarchical tree data structure from point cloud data.

The aim of this technique is to estimate the three values needed afterwards too. For each order of voxel, a list provides:

- the total number of points,

- the voxel ID, and
- the number of points that belong to that voxel.

This procedure continues up to the order in which the voxels are all empty. 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 3D output plot given is characterized by different colors, one for each order. The meaning of the different colors is the following one:

- cyan is used for the points classified as belonging to voxel 1°,
- black is used for the points classified as belonging to voxel 2°,
- yellow is used for the points classified as belonging to voxel 3°,
- blue is used for the points classified as belonging to voxel 4°,
- red is used for the points classified as belonging to voxel 5°,
- green is used for the points classified as belonging to voxel 6°, and
- purple is used for the points classified as belonging to voxel 7°.

The next part contains the description of the application of the voxel procedure to the structure. The following figure shows the input point cloud.

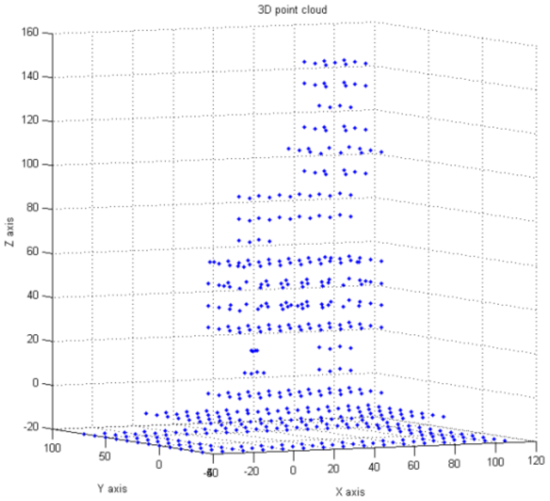


Figure 2.14 Input point cloud

The example body is composed of 469 points. The xmin is equal to - 40.000 m, the xmax is equal to 110.000 m, the ymin is equal to -40.000 m, the ymax is equal to 70.000 m, the zmin is equal to -20.000 m, and the zmax is 150.000 m.

Voxel 0° is composed of one cube and it contains all the 469 points.

Voxel 1° is subdivided into 8 cubes. The following table provides for each cube the number of points inside.

# cubes	1	2	3	4	5	6	7	8
# of points	99	98	98	97	8	35	8	26

Table 2.1 Voxel 1st order

Because none of the cubes are empty, the voxel subdivision will go on till the sixth order.
The 3D output plot given is characterized by different colors, one for each order. The meaning of the different colors is the same already mentioned.

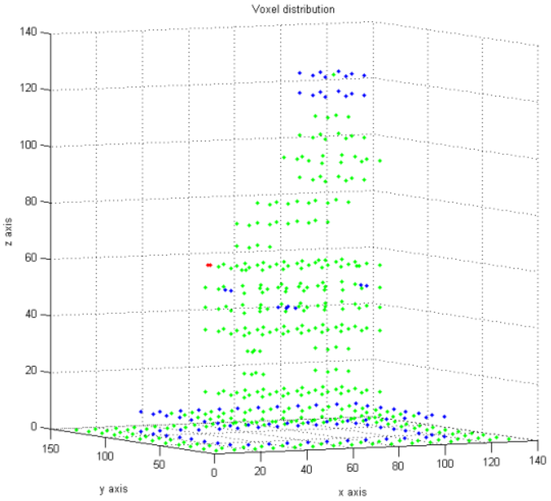


Figure 2.14 Voxel distribution

2.3. CLUSTER ANALYSIS

Clustering is the process of organizing objects into groups whose members are similar in some way [Anderberg, 1973, Dubes and Jain, 1988] and are dissimilar to objects belonging to other clusters. A simple graphical example is shown it the following figures. The former shows the unlabeled data and the points are in black.

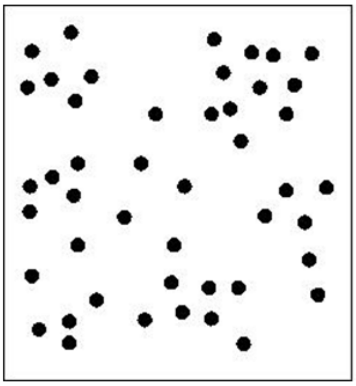


Figure 2.15 Starting points

The latter shows the data after the cluster analysis. Different colors represent the different clusters.

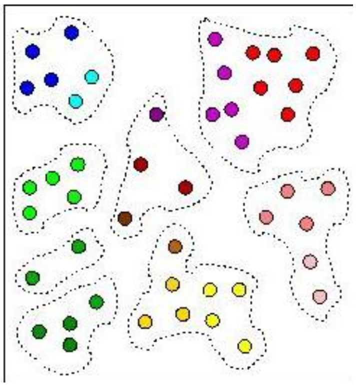


Figure 2.16 Clustered points

In this case the 8 clusters into which the data can be divided are easily identified using a similarity criterion (distance).

What constitutes a good clustering? For a successful outcome, the criterion used is dependent on the type of clustering that is sought after.

Clustering algorithms may be classified in differently; therefore different techniques can be used. They are the following:

- bottom – up,
- top – down,
- exclusive clustering, and
- overlapping clustering.

The first type of classification depends on the initial number of clusters. It may be equal to the number of data (bottom - up) or equal to one (top - down).

The second type of classification depends on the fact that a datum may belong to a single cluster (exclusive cluster) or to multiple clusters (overlapping clusters).

2.3.1. Bottom – up technique

The first type of cluster technique is the bottom – up technique. At the beginning all the points are classified as a single cluster. Then the algorithm merges the nearest clusters. This step will go on until the number of clusters reaches a prearranged value or till the minimum distance between the clusters does not exceed a fixed threshold.

2.3.2. Top – down technique

The second type of cluster technique is the top – down technique. At the beginning all the points are classified as belonging to the same cluster. Then the algorithm divides the starting cluster into different clusters. This step will go on until the number of clusters reaches a prearranged value.

2.3.3. Exclusive clustering

The third type of cluster technique is the exclusive technique.

Data are grouped in an exclusive way, so that if a certain datum belongs to a definite cluster then it could not be included in another cluster. A simple example of that is shown in the figure 2.17 below, where the separation of points is achieved by the straight line in black.

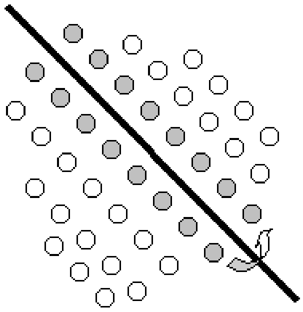


Figure 2.17 Exclusive clusters

This type of clustering algorithm is also known as hard clustering.

2.3.4. Overlapping clustering

The fourth type of cluster technique is the exclusive technique.

The overlapping clustering uses fuzzy sets to cluster data, so that each point may belong to two or more clusters with different degrees of membership. In this case, data will be associated to an appropriate membership value. A simple example of that is shown in the following figure, where the separation of points is achieved by overlapping circles.

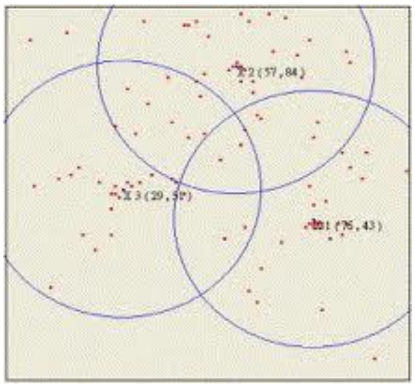


Figure 2.18 Overlapping clusters

2.3.5. Others

Another type of clustering is also known as soft clustering or fuzzy clustering.

The third type of classification depends on the type of algorithm used to separate the space. There are three types of clustering algorithms:

- partitioning,
- hierarchical, and
- density algorithms.

Each classification will be explained in detail in the following sections.

2.3.5.1. Partitioning algorithms

The partitioning algorithm typically starts with an initial partition of *D* and then uses an iterative control strategy to optimize an objective function. A two-step procedure is used. First, determine *k* representatives minimizing the objective function. Second, assign each object to the cluster with its representative “closest” to the considered object. The second step implies that a partition is equivalent to a Voronoi diagram and each cluster is contained in one of the Voronoi cells.

K-means clustering is a method in which each observation belongs to the cluster with the nearest mean.

The term "k-means" was first used by James MacQueen in 1967, though the idea goes back to Hugo Steinhaus in 1957. The standard algorithm was first proposed by Stuart Lloyd in 1957 as a technique for pulse-code modulation, though it was not published until 1982.

These centroids should be placed in a cunning way because a different location can cause a different result. So, the better choice is to place them as far away from each other as possible.

The following example shows 15 point that need to be divided into three subsets. The k initial "means" are randomly selected from the data set. In picture number 2.19 the 3 points are displayed with circles, the others with squares.

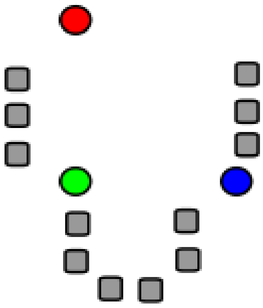


Figure 2.19 Three points chosen as centroids

K clusters are created by associating every observation with the nearest mean. The partitions represent the Voronoi diagram generated by the means. It is displayed in the following pictures.

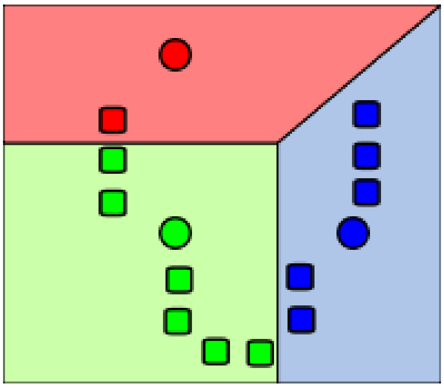


Figure 2.20 Three different areas

At this point k new centroids are re-calculated as barycenters of the clusters resulting from the previous step. In the following picture the new centroids are displayed with circles.

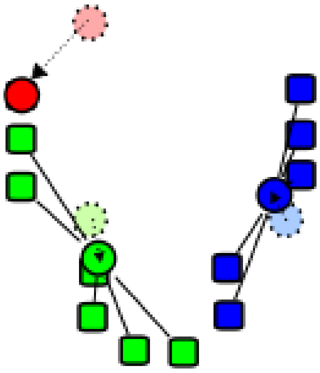


Figure 2.21 Three new centroids

After these k new centroids are found, a new binding has to be done between the same data set points and the nearest new centroid. The previous two steps are repeated until convergence has been reached. The result is displayed in figure 2.22.

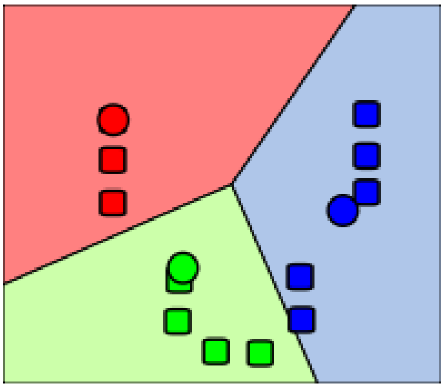


Figure 2.22 Final result

The k-means are represented with circles, the others with squares.

Although it can be proved that the procedure will always terminate, the k-means algorithm does not necessarily find the most optimal configuration, corresponding to the global objective function minimum. The algorithm is also significantly sensitive to the initial randomly selected cluster centres. The k-means algorithm can be run multiple times to reduce this effect.

2.3.5.2. Fuzzy C-means

This method was developed by Dunn in 1973 and improved by Bezdek in 1981.

Fuzzy c-means (FCM) is a method of clustering which allows one piece of data to belong to two or more clusters. This method is frequently used in pattern recognition. It is based on minimization of the following objective function:

$$J_m = \sum_{i=1}^{N} \sum_{j=1}^{C} u_{ij}^m \left\| x_i - c_j \right\|^2$$

where m is any real number greater than 1, u_{ij} is the degree of membership of x_i in the cluster j, x_i is the ith of d-dimensional measured data, c_j is the d-dimension center of the cluster, and $\|*\|$ is any norm expressing the similarity between any measured data and the center.

2.3.5.3. Quality Threshold

Quality Threshold is an algorithm that groups data into high quality clusters. Quality is ensured by finding large cluster whose diameter does not exceed a given user-defined diameter threshold. This method prevents dissimilar data from being forced under the same cluster.

The following steps provides the sequence that must be followed:

1. A random datum is chosen from the selected datum list.
2. The algorithm determines which datum has the greatest similarity to this gene. If their total diameter does not exceed the diameter threshold, then these two data are clustered together.
3. Other data that minimize the increase in cluster diameter are iteratively added to this cluster. This process continues until no datum can be added to this first candidate cluster without surpassing the diameter threshold.
4. A second candidate datum is chosen.
5. The algorithm determines which data has the greatest similarity to this second gene. All data in the selected gene list are available for consideration to the second candidate cluster.
6. Other data from the selected datum list that minimize the increase in cluster diameter are iteratively added to the second candidate cluster. The process continues until no datum can be added to this second candidate cluster without surpassing the diameter threshold.
7. The algorithm iterates through all data on the selected gene list and forms a candidate cluster with reference to each gene. In other words, there will be as many candidate clusters as there are data in the data list. Once a candidate cluster is formed for each data, all candidate clusters below the user-specified minimum size are removed from consideration.
8. The largest remaining candidate cluster, with the user-specified minimal number of datum member, is selected and retained as a QT cluster. The data within this cluster are now removed from consideration. All remaining data will be used for the next round of QT cluster formation.
9. The entire process is repeated until the largest remaining candidate cluster has fewer than the user-specified number of genes.
10. The result is a set of non-overlapping QT clusters that meet quality threshold for both size, with respect to number of data, and similarity, with respect to maximum allowable diameter.
11. Data that does not belong in any clusters will be grouped under the “unclassified” group.

2.3.5.4. DBSCAN

DBSCAN (for density-based spatial clustering of applications with noise) is a data clustering algorithm proposed by Martin Ester, Hans-Peter Kriegel, Jörg Sander and Xiaowei Xu in 1996. It is a density-based clustering algorithm because it finds a number of clusters starting from the estimated density distribution of corresponding nodes.

When looking at the sample sets of points depicted in figure 2.23, it is possible to detect clusters of points and noise points not belonging to any of those clusters.

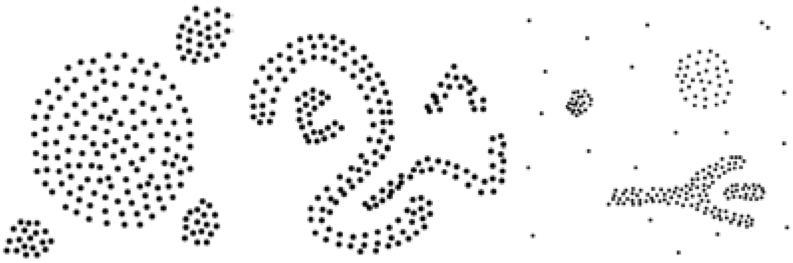


Figure 2.23 Samples

The main reason why it is possible to recognize the clusters is that within each cluster, a typical density of points is considerably higher than outside of the cluster. Furthermore, the density within the areas of noise is lower than the density in any of the clusters.
Different colors in the following picture represent the different clusters.

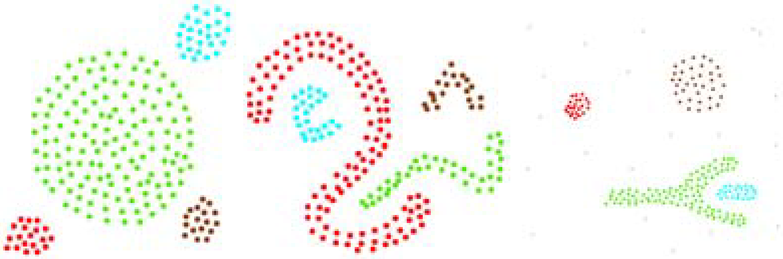


Figure 2.24 Clustering discovered by DBSCAN

A cluster obtained with DBSCAN technique satisfies the following two properties:

1. all points within the cluster are mutually density-connected and
2. if a point is density-connected to any point of the cluster, it is part of the cluster as well.

DBSCAN requires two parameters:

- ε and
- minPts: the minimum number of points required to form a cluster [Wikipedia].

It starts with an arbitrary starting point that has not been visited. This point's ε -neighborhood is retrieved, and if it contains sufficiently many points, a cluster is started. Otherwise, the point is labeled as noise. Note that this point might later be found in a sufficiently sized ε -environment of a different point and hence becomes part of a cluster.
If a point is found to be a dense part of a cluster, its ε -neighborhood is also part of that cluster. Hence, all points that are found within the ε -neighborhood are added, as is their own ε -neighborhood when they are also dense. This process continues until the density-connected cluster is completely found. Then, a new unvisited point is retrieved and processed, leading to the discovery of a further cluster or noise.

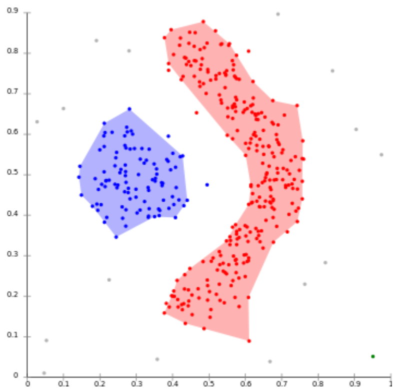


Figure 2.25 DBSCAN can find non-linearly separable clusters. This dataset cannot be adequately clustered with k-means or EM clustering.

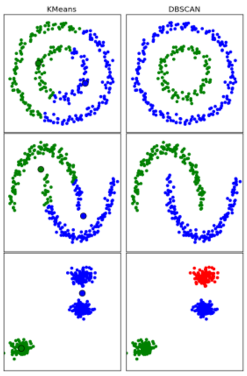


Figure 2.26 Another comparison between k-means and DBSCAN.

The adopted cluster technique is a top – down and exclusive one. Different methods are used, depending on the different types of input point cloud.
It is possible to apply the cluster analysis in three different ways:

- to cluster the data vertically,
- to cluster the data horizontally, and
- to cluster to data depending on the characteristic of the points themselves inside the point cloud.

The first kind of cluster analysis is applied using the first parameter estimated using the "voxel" procedure. For each level, the information is listed in this way:

- vertical cluster ID,
- number of points belonging to that vertical cluster,
- height, and
- ID point at that level.

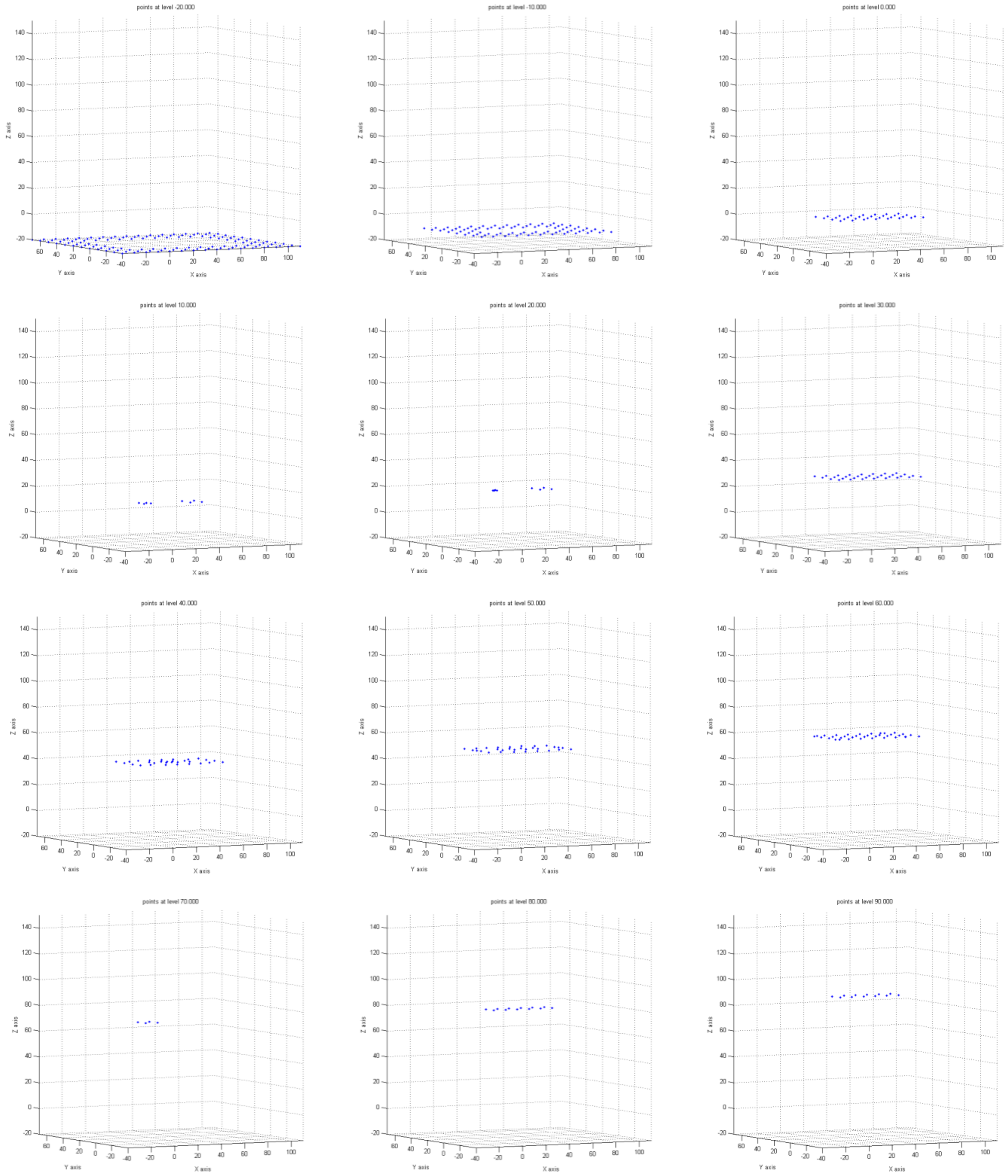
The following table provides an example of the output:

Vertical cluster ID	# of points	Height (m)	Point IDs		
1	132	-20.000	401	402	403

			…		
			60	87	68
…					
52	8	150.000	272	277	276
			…		
			279	278	275

Table 2.5 Vertical clusters

The next part contains the description of the application of the cluster analysis procedure to the structure. One 3D plot for each vertical cluster is provided and one 3D plot for the input point cloud. The former are displayed in figure 2.27 and the latter in figure 2.28.



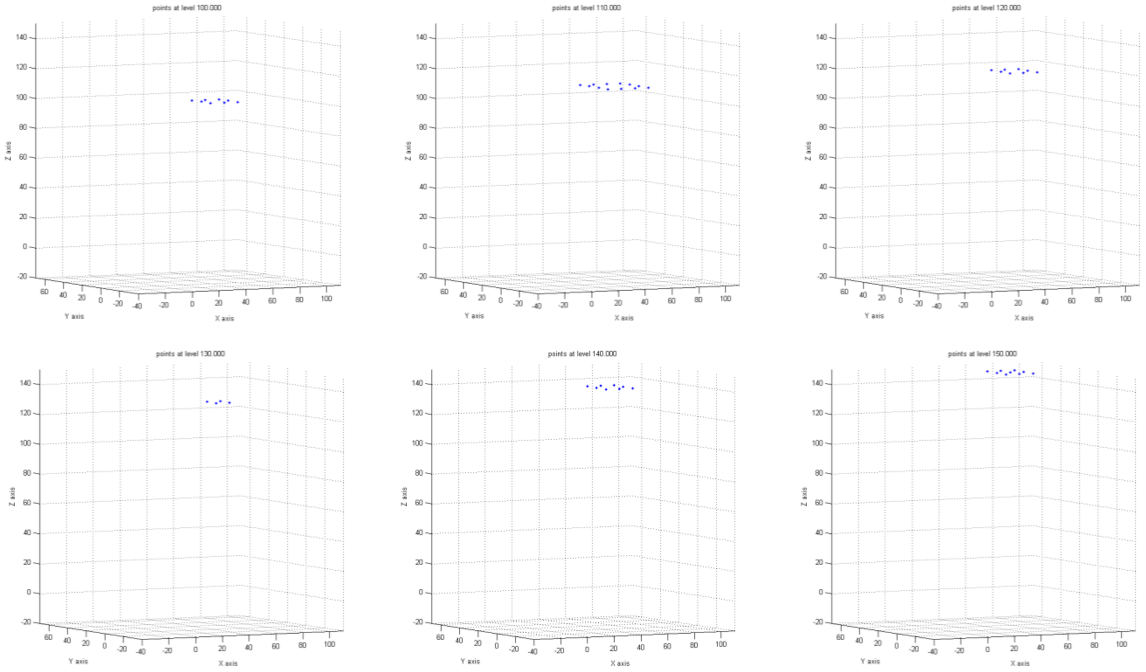


Figure 2.27 Vertical clusters

The following picture displays the 3D plot containing all the vertical clusters. The alternation of colors from red to blue represents the switch from one vertical cluster to the next.

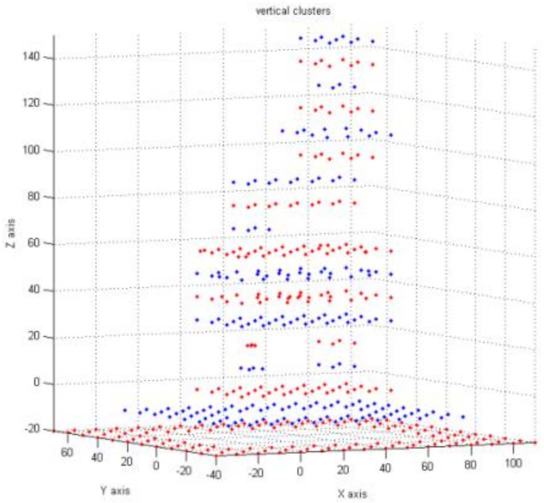


Figure 2.28 Vertical clusters

There are 18 vertical clusters.

The second kind of cluster analysis is applied using the second parameter estimated using the “voxel” procedure. 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.

Each horizontal cluster is described by:

- horizontal cluster ID,

- its height,
- the point ID belongs to the horizontal cluster, and
- the number of points that belong to the horizontal cluster.

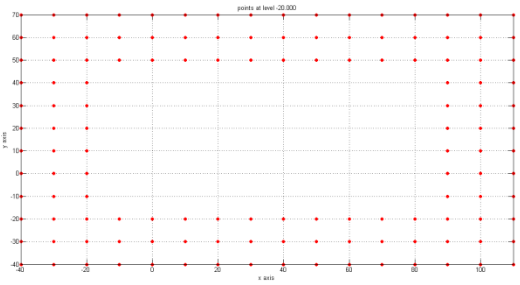
The following table provides an example of the output:

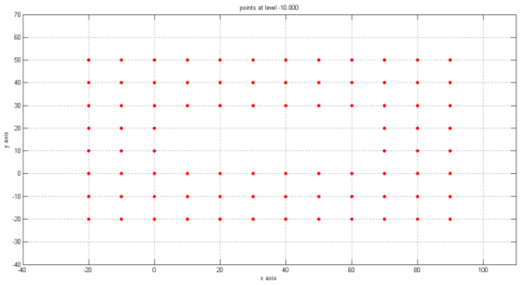
Horizontal cluster ID	Height (m)	Point IDs			# of point
1	-20.000	401	402	413	132
		476	519	525	
…					
22	150.000	272	277	276	9
		278	279	285	

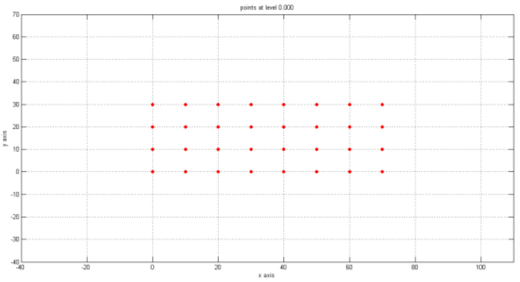
Table 2.6 Horizontal clusters

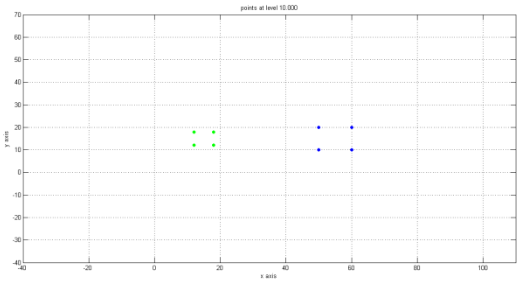
The number of 3D plots is equal to the number of vertical clusters. In fact, if a vertical cluster contains more than one horizontal cluster, the 3D plot contains all the horizontal clusters at that level.

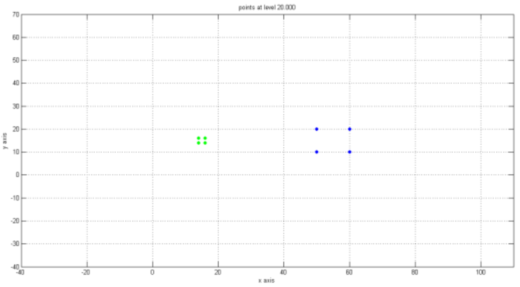
The following pictures show the horizontal clusters. If the vertical cluster contains one horizontal cluster, the points are displayed in red. If the vertical cluster contains more than one horizontal cluster, the points are displayed with different colors, one for each horizontal cluster.

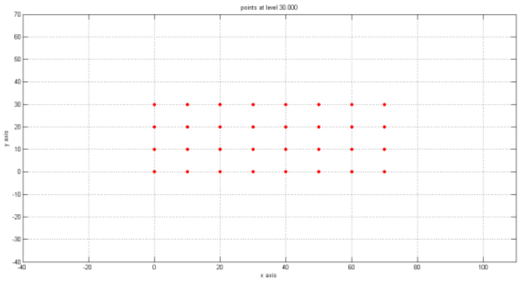


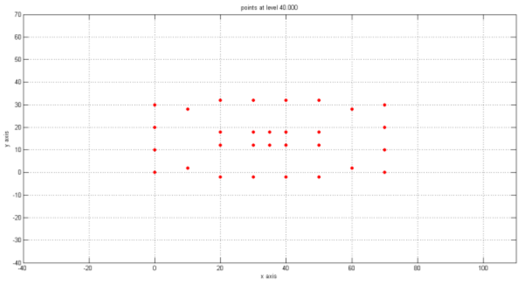


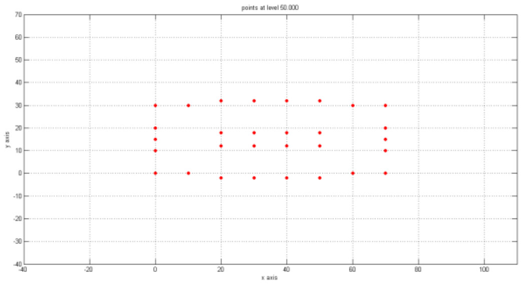


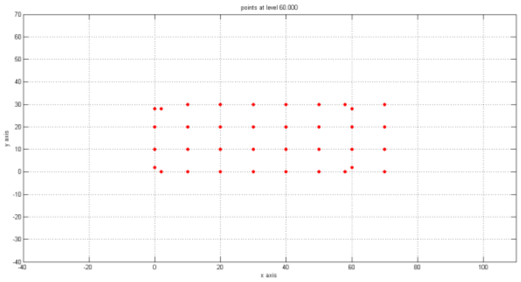


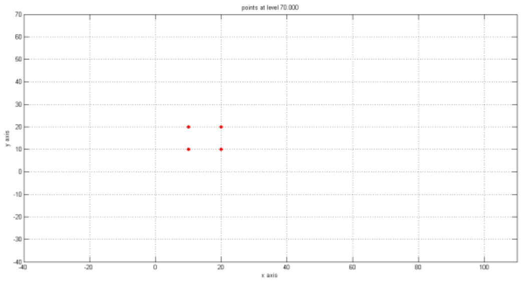


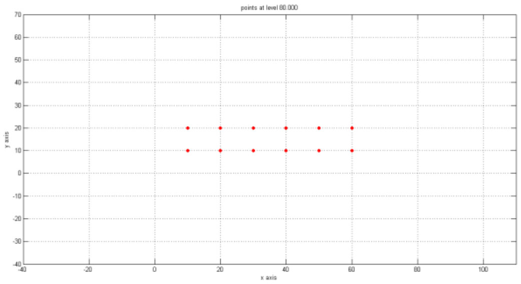


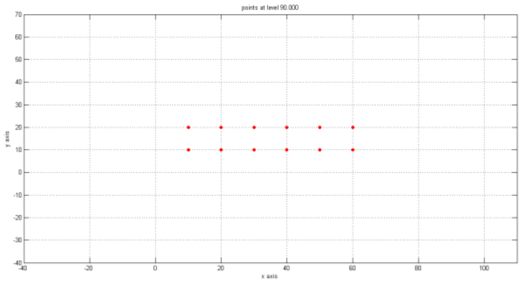

points at level 40.000
70
60
50
40
30
20
10
0
-10
-20
-30
-40
y axis
-40
-20
0
20
40
60
80
100
x axis

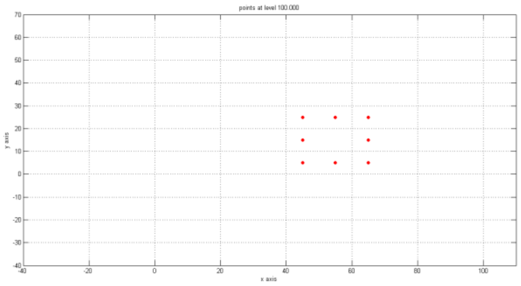

points at level 50.000
70
60
50
40
30
20
10
0
-10
-20
-30
-40
y axis
-40
-20
0
20
40
60
80
100
x axis

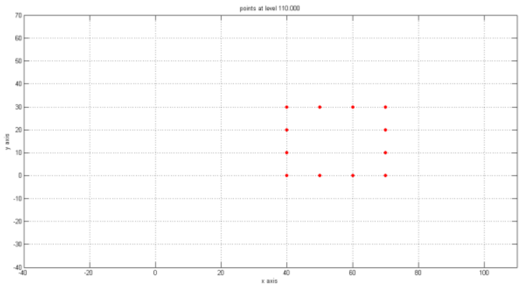

points at level 60.000
70
60
50
40
30
20
10
0
-10
-20
-30
-40
y axis
-40
-20
0
20
40
60
80
100
x axis

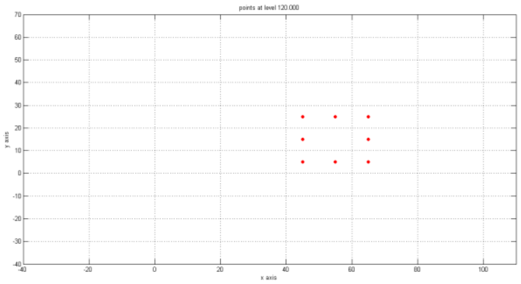

points at level 70.000
70
60
50
40
30
20
10
0
-10
-20
-30
-40
y axis
-40
-20
0
20
40
60
80
100
x axis

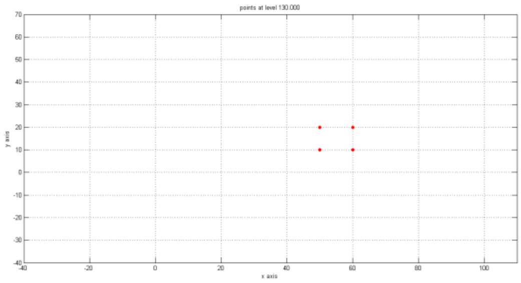

points at level 80.000
70
60
50
40
30
20
10
0
-10
-20
-30
-40
y axis
-40
-20
0
20
40
60
80
100
x axis

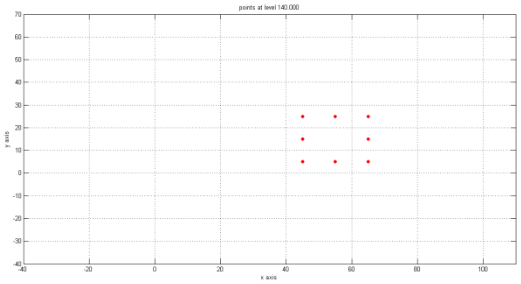

points at level 90.000
70
60
50
40
30
20
10
0
-10
-20
-30
-40
y axis
-40
-20
0
20
40
60
80
100
x axis


points at level 100.000
70
60
50
40
30
20
10
0
-10
-20
-30
-40
y axis
-40
-20
0
20
40
60
80
100
x axis


points at level 110.000
70
60
50
40
30
20
10
0
-10
-20
-30
-40
y axis
-40
-20
0
20
40
60
80
100
x axis


points at level 120.000
70
60
50
40
30
20
10
0
-10
-20
-30
-40
y axis
-40
-20
0
20
40
60
80
100
x axis


points at level 130.000
70
60
50
40
30
20
10
0
-10
-20
-30
-40
y axis
-40
-20
0
20
40
60
80
100
x axis


points at level 140.000
70
60
50
40
30
20
10
0
-10
-20
-30
-40
y axis
-40
-20
0
20
40
60
80
100
x axis

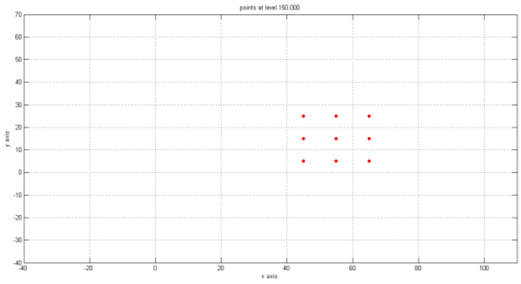

points at level 150.000
70
60
50
40
30
20
10
0
-10
-20
-30
-40
y axis
-40
-20
0
20
40
60
80
100
x axis

Figure 2.29 Horizontal clusters

In this study case there are 22 horizontal clusters.

The third kind of cluster analysis is applied to data depending of their characteristics. As mentioned earlier, in chapter 7 I will explain how it is possible to obtain the 3D model of an airport. It is necessary to cluster the input point cloud into different sets, depending on what the acquired points represent from an actual airport.

In the airport study case, figure 2.30 shows the image “InputSets.tif” with two different colors: the center line (first set) is in red and the other points (second set) are in blue.

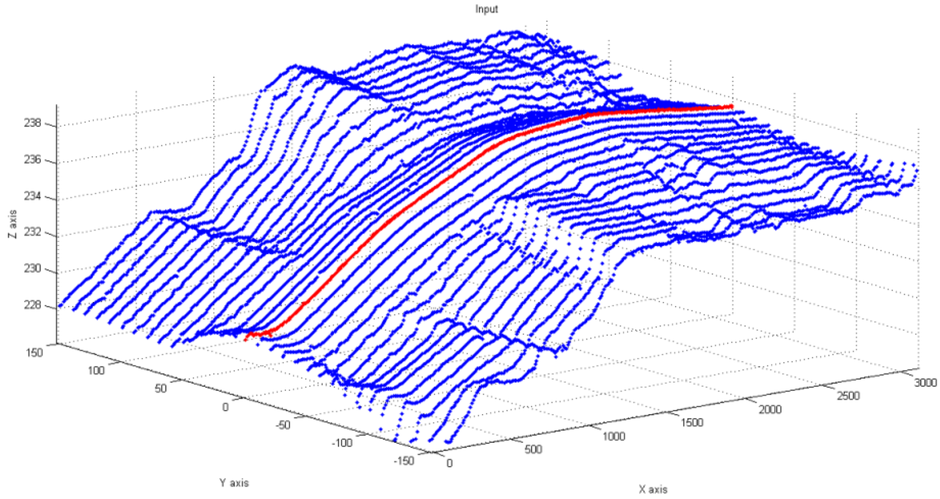


Figure 2.30 Center line in red and the other points in blue

The dimension of each part depends on the Aerodrome Reference Code. This code prescribes the widths and the cross slopes. Through this information, it is possible to cluster the input point cloud. Figure 2.31 shows the results of the cluster analysis.

- Red is used for the points classified as belonging to the center line cluster,
- Blue is used for the points classified as belonging to the runway cluster,
- Green is used for the points classified as belonging to the runway shoulders cluster,
- Black is used for the points classified as belonging to the Clear and Grated Area cluster,
- Purple is used for the points classified as belonging to the runway strip cluster, and
- Yellow is used for the points classified as belonging to the last cluster.

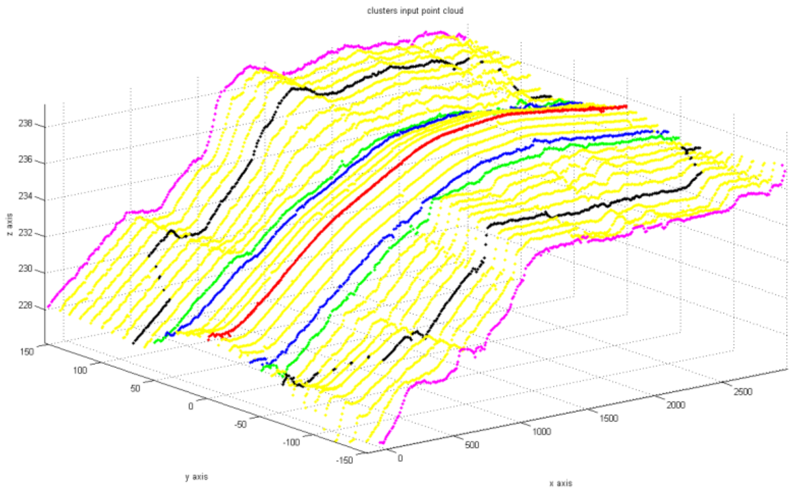


Figure 2.31 Clustered input point cloud

The following table provides an example of the output:

Point ID	X	Y	Height (m)	Cluster ID
…				
4942	90.34692125	89.54085275	228.53600000	2
4943	101.52819376	89.81579838	228.654	2
…				

Table 2.7 Cluster analysis

As mentioned, I will explain how it is possible to obtain the 3D model of an aquifer in chapter 8. It is necessary to cluster the input point cloud into different sets.
The aquifer is divided into four layers: this information is used to divide the data into clusters. The following image shows the clusters using four different colors:

- green for the points classified as belonging to the land surface layer,
- red for the points classified as belonging to the aquitard up layer,
- light blue for the points classified as belonging to the aquitard down layer, and
- blue for the points classified as belonging to the aquiclude layer.

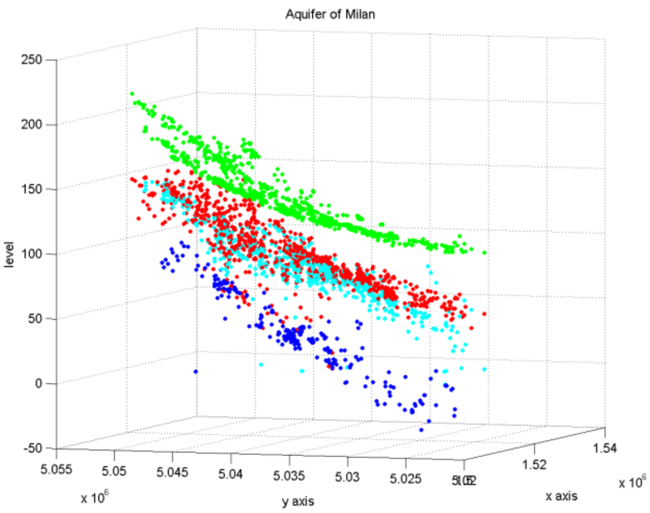


Figure 2.32 Cluster analysis of the aquifer

2.4. FEATURE EXTRACTION

It is possible to extract different types of information from the point cloud. Also, it is possible to extract the perimeter of each horizontal cluster and to classify the type of each horizontal cluster.

It is necessary to obtain the perimeter in order to know where it is necessary to interpolate and where to extrapolate information.

There are different kinds of perimeters; here I will present two of them. The first is the classic one and the second is the piecewise Catmull – Rom's lines. The classic one is used in case the study case is characterized by rough edges; the piecewise Catmull – Rom's lines in the opposite case.

There are different types of horizontal cluster. Here the attention is focused on the classification based on the distinction between sowns and chains.

A sown is the representation of a horizontal plane formed by dense points. A chain is a planar path modeled by rare points. Figure 2.33 shows a chain with blue dots and a sown with red dots.

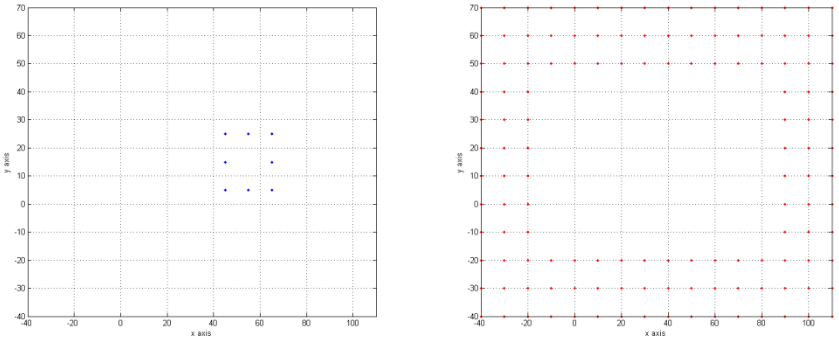


Figure 2.33 Chain (left) and sown (right)

The following table provides an example of the output:

Horizontal cluster ID	Height (m)	Type
9	40.000	Chain

10	40.000	Chain
11	50.000	Chain
…		

Table 2.8 distinction between chains and sowns

This distinction is done because the different type of horizontal cluster affects both the relational matching and the Delaunay triangulation.

I will explain the relational matching in section 5 and the Delaunay triangulation in section 6. Here I will explain the connection between the type of horizontal cluster and the two techniques.

If the relational matching is done between a chain and a sown, the number of matched points is different because a chain and a sown are composed of a different number of points.

If the relational matching is carried on between two chains, it connects each point of the former one with each point of the latter one.

A brief introduction about the perimeter technique will be followed by a detailed description of the two different kinds of perimetrations.

2.4.1. Perimeter

In order to obtain a continuous 3D model, all the points that belong to each cluster must be perimetred with a closed loop.

A perimeter is a path that surrounds an area or a path that connects a group of points. The word comes from the Greek *peri* (around) and *meter* (measure).

The generation of closed loops is easy only in elementary cases. It becomes more complex as the shape of the loop becomes more complicated. In order to implement this part of the research, the reconstruction of a four letters (F, G, R, and W) and a spider shape was preliminarily performed, starting from the raw points.

The next images show the shapes analyzed.

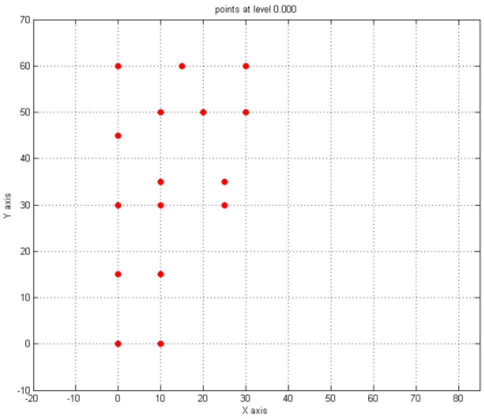


Figure 2.34 Point cloud for letter F

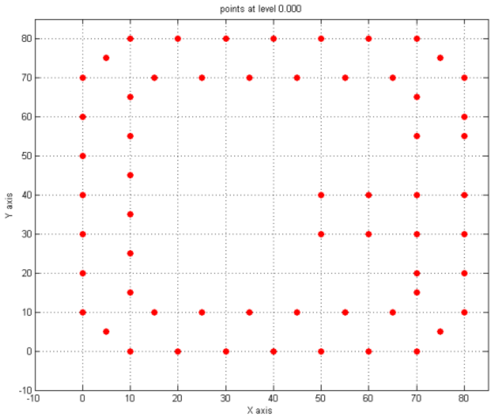


Figure 2.35 Point cloud for letter G

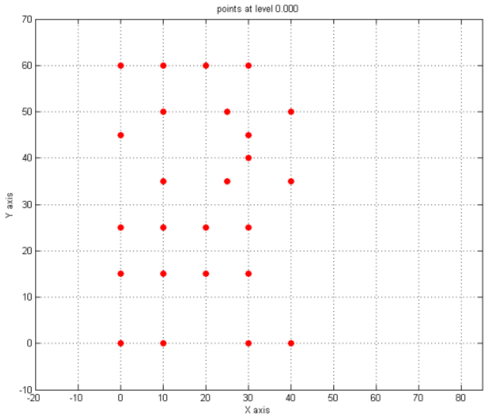


Figure 2.36 Point cloud for letter R

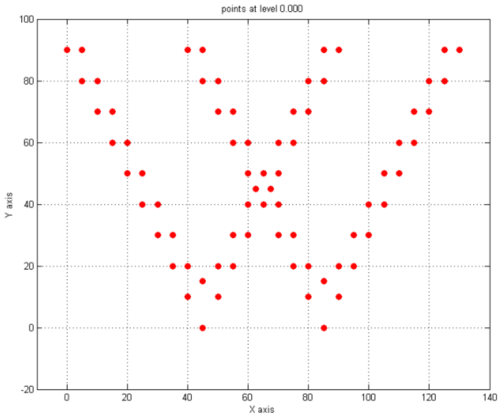


Figure 2.37 Point cloud for letter W

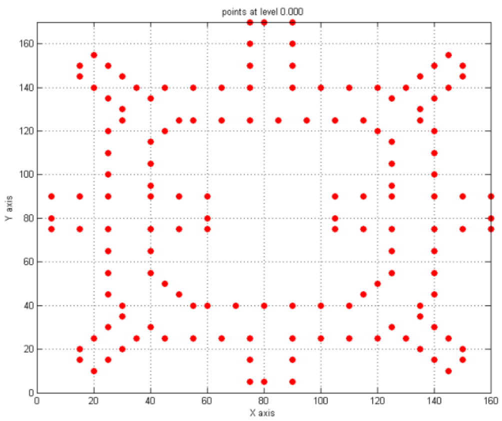


Figure 2.38 Point cloud for a spider shape

To find a closed loop, first the software connects a point to its closest point; next it connects this second point to its closest (excluding the first) and so forth (in order to keep a unique sense of advancement). 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, and
- 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.

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 reach the root starting from the farthest leaf.

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.

Figure 2.39 shows the 14 input points.

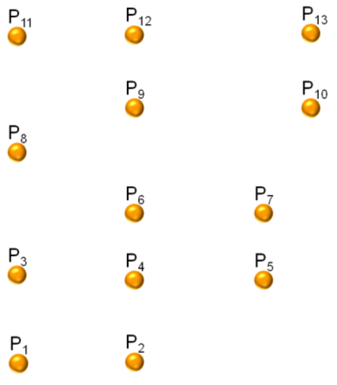


Figure 2.39Input points

The solution showed in figure 2.40 is wrong because the perimeter is not closed. The closed loop is a path whose initial point is equal to the terminal point. Contour #1 starts in P_1 and finishes in P_2. It is therefore wrong.

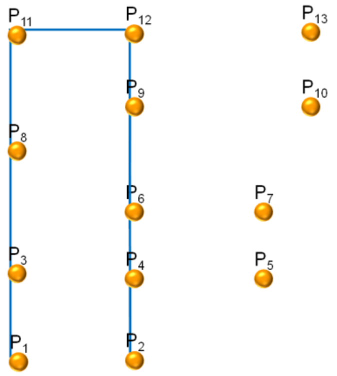


Figure 2.40 Contour #1

All the points must be taken. Contour #2 does not contain P_5, P_8, P_{11}, and P_{14}.

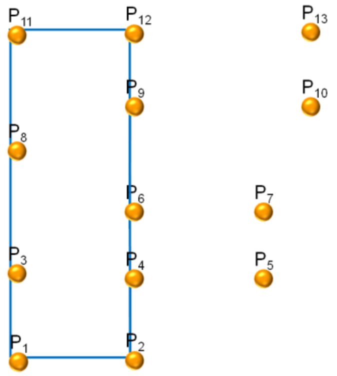


Figure 2.41 Contour #2

All the points must be taken only once. Contour #3 takes P_4, P_6, P_8, P_9, and P_{10} at least two times. It is therefore wrong.

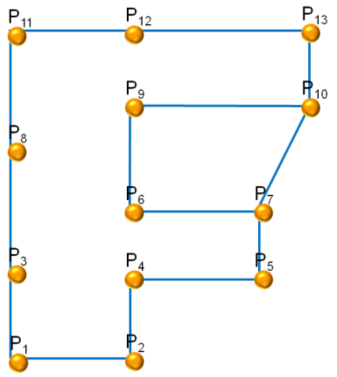


Figure 2.42 Contour #3

All sides, if taken, only once and they do not have to cross each other. Contour #4 is therefore wrong.

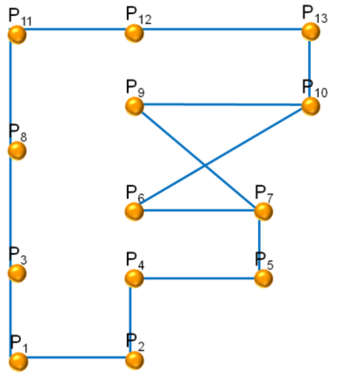


Figure 2.43 Contour #4

Contour #5 is the correct one.

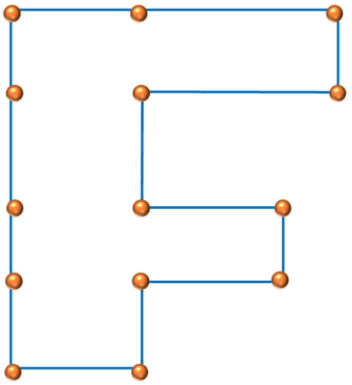


Figure 2.44 Contour #5

The follow three images show the closed loops of the other letters analyzed.

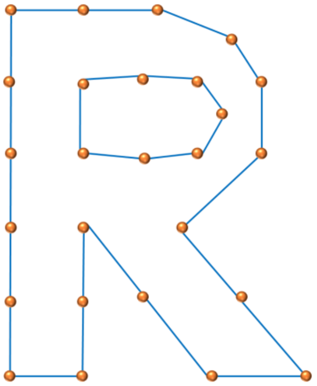


Figure 2.45 Contour, letter R

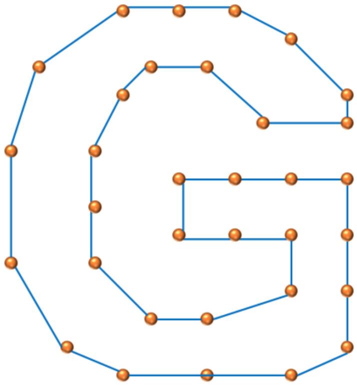


Figure 2.46 Contour, letter G

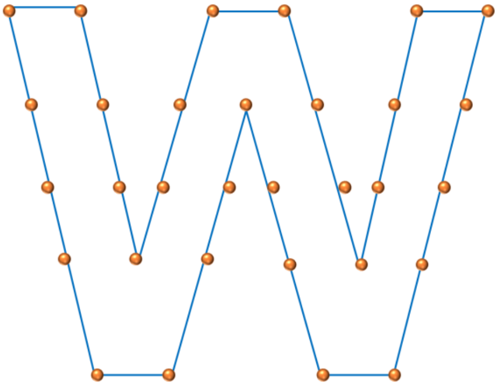


Figure 2.47 Contour, letter W

There are two different types of closed loops. The first is the piecewise Catmull – Rom's lines, the second type is the classic one. The following two paragraphs contain a description of the two different techniques.

2.4.2. 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 an example of 5 input points.

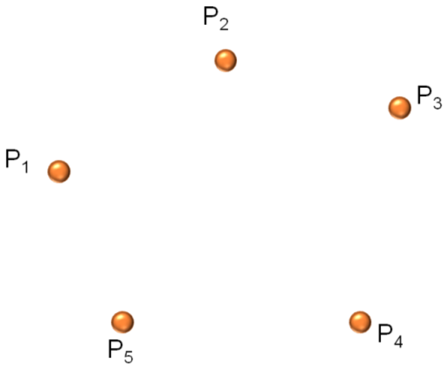


Figure 2.48 Five input points

The purpose is to connect these five points with piecewise Catmull – Rom lines.

Starting from points P_1, P_2 and P_3, the straight line (in light blue in figure 2.49) is parallel to the line joining the points P_1 and P_3.

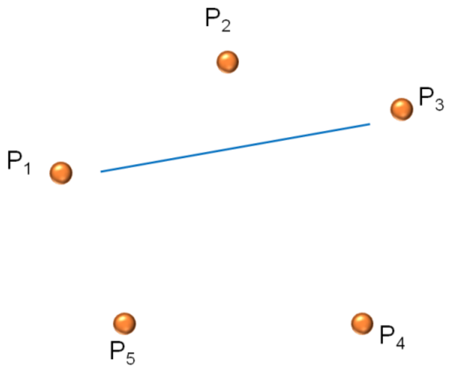


Figure 2.49 First line of the Catmull – Rom closed loop

The first straight line passes through point P_2 and it is parallel to the line joining the point P_1 and P_3. See figure 2.50 for a better representation.

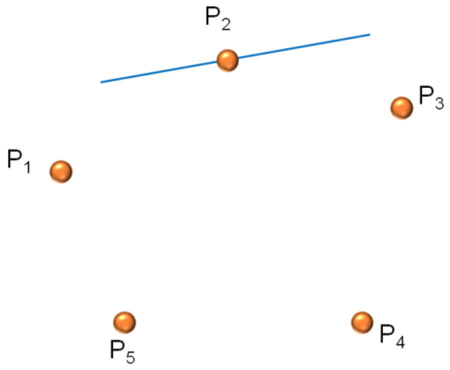


Figure 2.50 First line of the Catmull – Rom closed loop

The same method is applied to the other four sides. The perimeter is obtained. This image shows the Catmull-Rom closed path between the 5 input points.

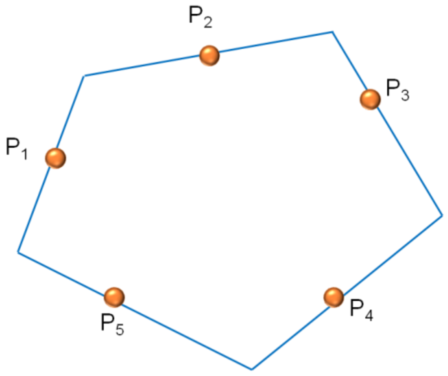


Figure 2.51 Perimeter with Catmull – Rom

The advantage of this type of interpolation is the possibility to extrapolate other information.
It is possible to evaluate:

- the two coordinates of each new points (in blue in figure 2. x) and
- for each straight line (in orange in figure 2.x):
  - type of straight line:
    - $y = ax + b$
    - $x = cy + d$
      - in the first case, a y is displayed
      - in the second, an x is displayed
  - the slope of the straight line and
  - y-intercept or x-intercept.

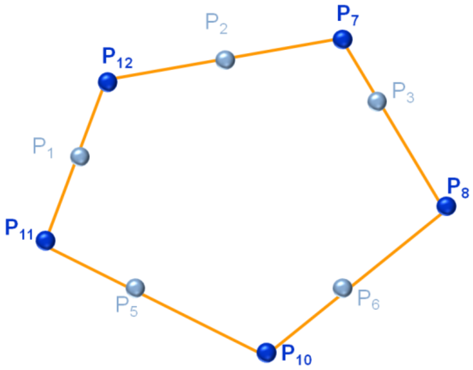


Figure 2.52 Catmull – Rom closed loop (in orange), the five input points (in light blue) and the five new points (in blue)

The first table refers to the new points (in blue in the picture 2.52). The second table refers to the sides (in orange in the picture 2.52).

Points					
Interpolation ID	**Horizontal cluster ID**	**Point ID**	**X**	**Y**	**Z**
1	1	5001	0.000	0.000	0.000
…			…	…	…

Table 2.9 Points

Lines								
Interpolation ID	**Horizontal cluster ID**	**Side ID**	**Point ID**			**Type of line**	**Slope**	**x-intersect or y-intersect**
1	1	1	1	2	3	y	0.000	0.000
		2	2	3	4	y	0.000	0.000
	…	…	…	…	…	…	…	…

Table 2.10 Lines

The piecewise Catmull-Rom lines perimeter is used in case the structure is characterized by smooth edges. An example of this kind of structures is designed by the architect Zaha Hadid. An example of her futuristic building is the Defunct Factory in downtown Belgrade and a picture of this building is in the following figure.

Figure 2.53 Beko Building, Zaha Hadid

If this perimetration is set, the following step uses Bézier splines. They will be explained in the appendix.

2.4.3. 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 2.53.

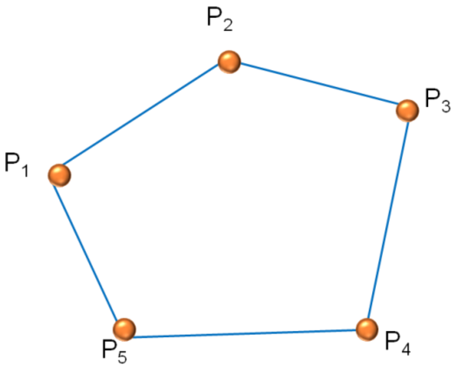


Figure 2.53 Classic perimeter

The Delaunay triangulation will be explained in subchapter 6. It is here mentioned because it is possible to extract the close loop from the Delaunay triangulation if the structure is convex.

In order to do this, a short example will clarify the adopted procedure. The following picture (Figure 2.54) shows an example of Delaunay triangulation between eight points. There are fourteen sides.

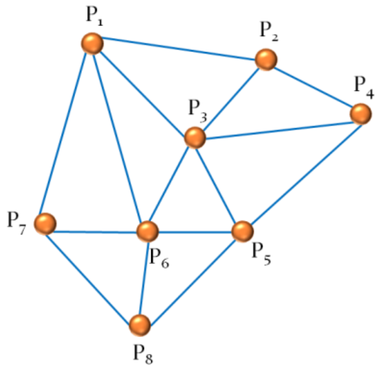


Figure 2.54 Triangulation

Each triangle is composed of three sides. If a side is in common with more than one triangle, it is deleted. On the contrary, the side is classified as a side of the closed path.

The sides $P_1 - P_3$, $P_2 - P_3$, $P_4 - P_3$, $P_5 - P_3$, $P_5 - P_6$, $P_8 - P_6$, $P_7 - P_6$, $P_1 - P_6$, and $P_6 - P_3$ belong to more than one triangle. These sides are deleted and the contour is obtained. The following picture (Figure 2.55) shows the result.

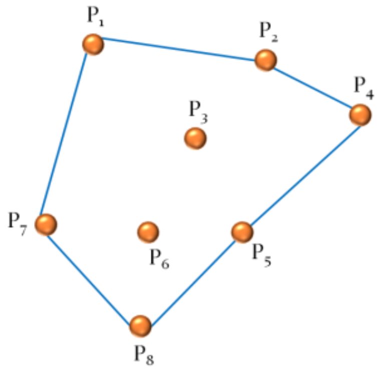


Figure 2.55 Perimetration

It is not possible to obtain the closed loop starting from the Delaunay triangulation if the body/structure/horizontal cluster is concave.

Figure 2.56 provides the 14 points that represent the “F” and its triangulation.

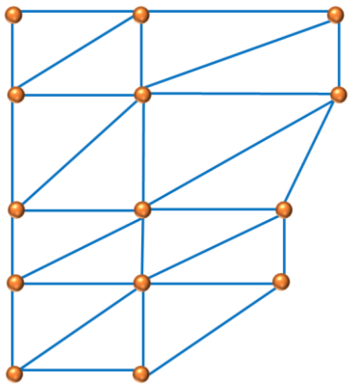


Figure 2.56 Delaunay triangulation

Figure 2.57 shows the perimetration obtained by deleting the sides in common.

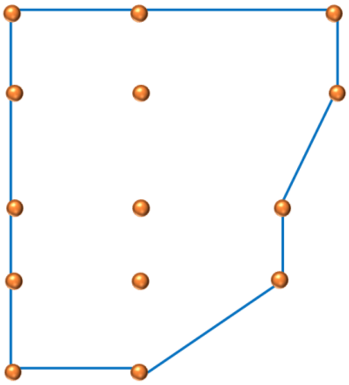


Figure 2.57 Perimetration

Figure 2.58 shows the correct perimetration.

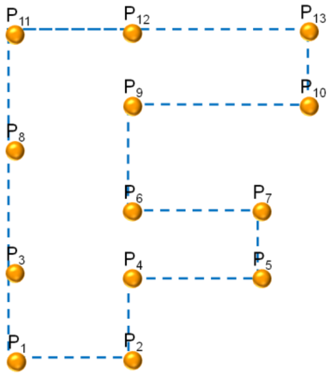


Figure 2.58 Perimetration

2.4.4. Type: chain or sown?

As mentioned, it is necessary to classify a horizontal cluster in a chain cluster or a sown cluster. In fact, a structure can be composed of two different sets of information: sowns and vertical or almost vertical walls.

The information is provided in the following way:

- horizontal cluster ID,
- type of clustres, and
- height.

The following table provides the classification of each horizontal cluster.

Horizontal cluster ID	Type	Height (m)
1	sown	-20.000
2	sown	-10.000
3	sown	0.000
4	chain	10.000
5	chain	10.000
6	chain	20.000
7	chain	20.000
8	sown	30.000

9	chain	40.000
10	chain	40.000
11	chain	50.000
12	chain	50.000
13	sown	60.000
14	chain	70.000
15	sown	80.000
16	sown	90.000
17	chain	100.000
18	chain	110.000
19	chain	120.000
20	chain	130.000
21	chain	140.000
22	chain	150.000

Table 2.11 Feature extraction

Figure 2.60 displays the point cloud with two colors: green if the horizontal cluster represents a sown, blue if it is classified as a chain.

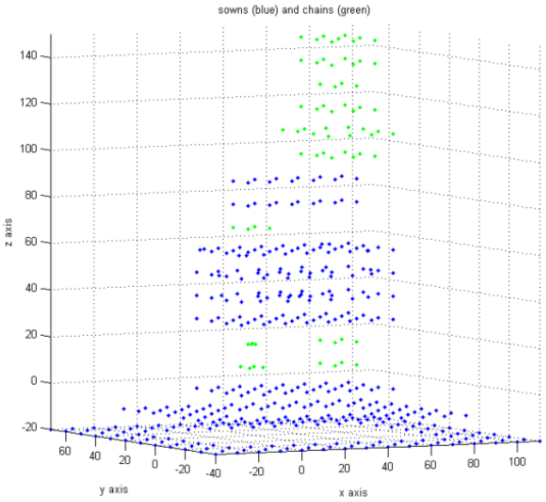


Figure 2.60 Chains (in blue) and sowns (in green)

2.5. RELATIONAL MATCHING

Relational matching is a method for finding the best correspondences between data to generate a vertical boundary.
Matching algorithms may be analyzed by posing three fundamental questions:

- what kind of data is matched?
- what is the best match?
- how to find the best match? [Lecture notes in computer science, 628]

The kind of data that has to be matched, are points that belong to different but consecutive clusters.

The relational matching method is used to search the correct correspondences between points that belong to consecutive layers to generate a vertical boundary. Figure 2.61 shows two sets of points that belong to different but consecutive levels in orange and the two perimeters in light blue.

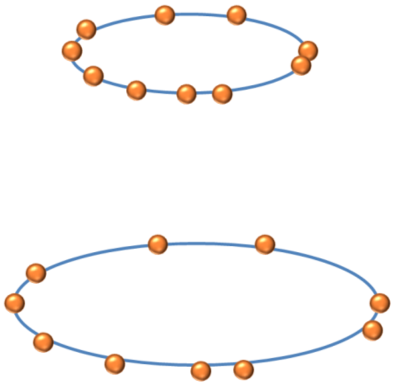


Figure 2.61 Points and contours

The first proposed solution in figure 2.62 is wrong and the consistency of matched sides is missing.

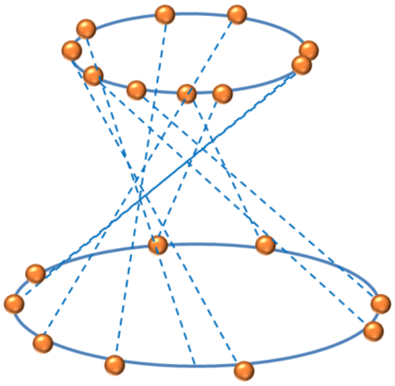


Figure 2.62 First solution

In order to obtain the best match, the connections must satisfy the following properties:

- the points in question must be as close as possible while still belonging to consecutive clusters,
- if points belong to the same vertical, they are easily matched, and
- if not, the software looks to create connections along the same direction [Forcella, Mussio, Reconstructing CCTV 2011].

The above mentioned “same direction” can be almost vertical or almost horizontal. I will provide two examples to create a better understanding of the technique and the results.
Picture 2.63 shows the correct relational matching between the 20 points showed in figure 2.47. These points are connected using the vertical direction.

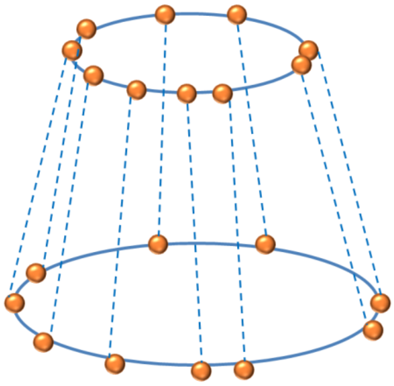


Figure 2.63 Points, contours and relational matching

The relational matching is applied to obtain the delimitation of the input point cloud and the starting point for Delaunay triangulation if the input point cloud represents a structure. The relational matching is also applied to obtain the diagram of the areas automatically if the input point cloud represents an infrastructure. To do so, Delaunay triangulation must be applied previously.

The relational matching is applied if to obtain the delimitation of the input point cloud the input point cloud represents an aquifer.

Delaunay triangulation is applied after the relational matching. In fact, the relational matching generates the so called anti-prisms.

The following picture displays an example. The input points in orange, the perimeter in light blue, and the relational matching in dashed green lines.

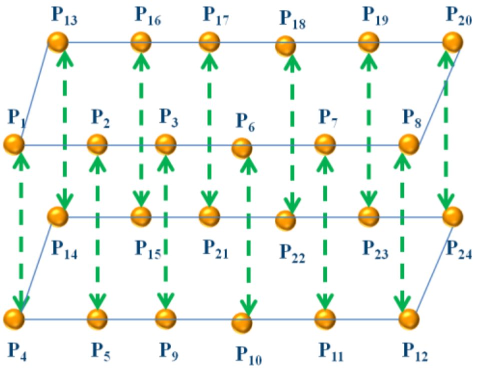


Figure 2.64 Relational matching

The triangulation will start from the sides generated by relational matching and the perimetration. The following picture show the previous example with the input point in orange, the perimetration in light blue, the relational matching in green and the beginning of the triangulation in red.

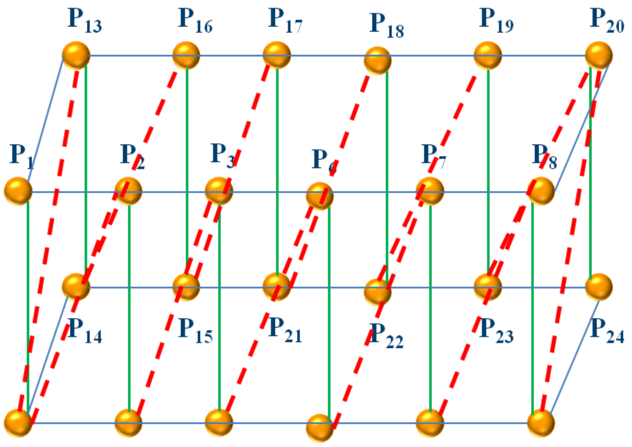


Figure 2.65 Relational matching and the beginning of the triangulation

As mentioned, the “same direction” can be almost horizontal or almost vertical. These types of relational matching are used if to obtain the Brückner profiling automatically, the input point cloud is an airport. To get it, the diagram of the areas is required. The relational matching is used to this achieve the goal.

The horizontal relational matching is done twice: the first time it is applied to the point cloud, the second time to the 3D model. The relational matching is displayed in the following picture. The points are displayed in orange, the point IDs in blue, and the relational matching in red dashed arrows.

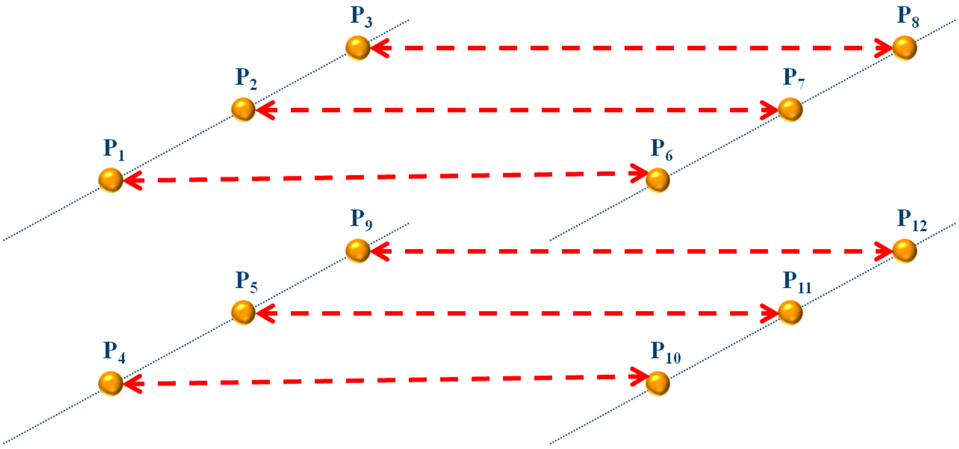


Figure 2.66 Relational matching in horizontal

This type of relational matching is repeated for the length of the airport. In this study case the length is equal to 300 meters and the space parameter is equal to 10 meters so the relational matching is done 30 times for the point cloud and 30 times for the model.

The vertical relational matching connects the points belonging to the point cloud and the points belonging to the model. An example of the relational matching is displayed in the following picture. The points are displayed in orange, the point IDs in blue, and the relational matching in green dashed arrows.

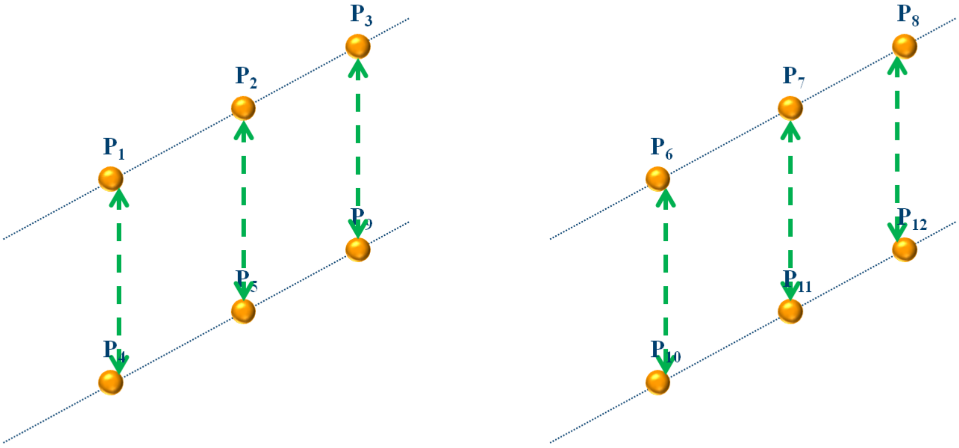


Figure 2.67 Vertical relational matching

This type of relational matching is repeated for the length of the airport. In this study case the relational matching is done 31 times.

As mentioned, these two types of relational matching are used to obtain the diagram of the area. It is achieved evaluating the area that is included between points connected using the relational matching. The following picture will clarify the method. The points are displayed in orange, the point IDs in blue, the relational matching in vertical in green dashed arrows, the relational matching in horizontal in red, and the area is highlighted in light blue.

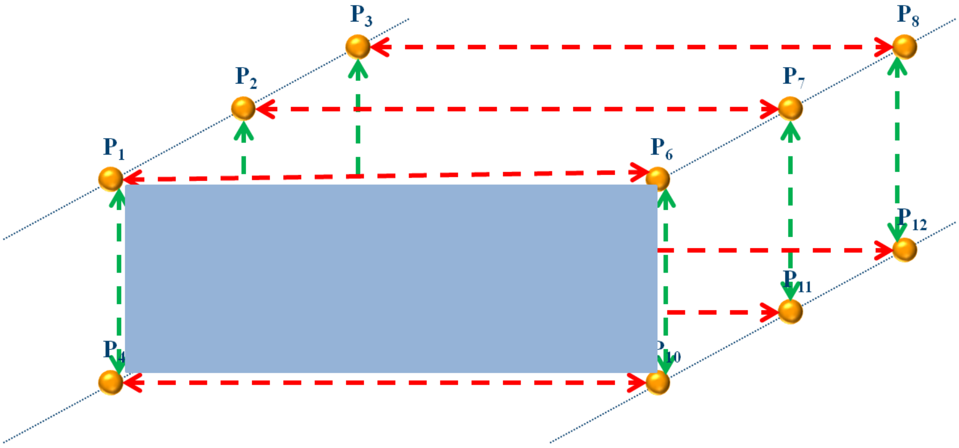


Figure 2.68 Relational matching to get the diagram of the areas

The value of area in light blue is calculated using Delaunay triangulation and the stabilized Erone formula, as shown in the following picture. I will explain Delaunay triangulation in the next subchapter.

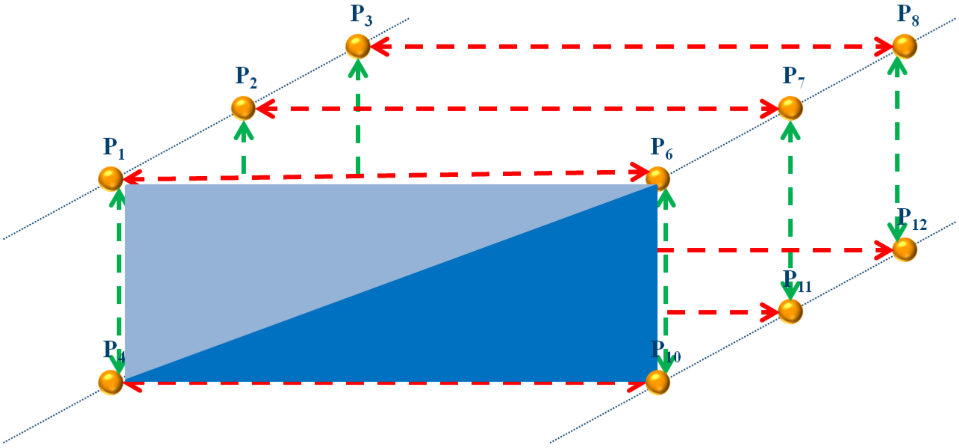


Figure 2.69 Relational matching after Delaunay triangulation

2.6. DELAUNAY TRIANGULATION

It is possible to connect a set of points with triangles. This method is called triangulation. If the acquired information is two-dimensional, the triangulation generates a 2D field. An example of this triangulation is the following image.

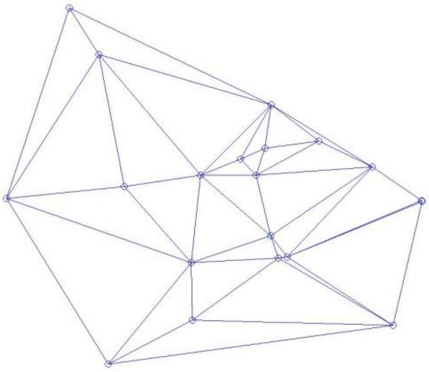


Figure 2.70 2D triangulation

If the input information is three-dimensional, the triangulation generates a 3D field. An example of the plot is the following plots.

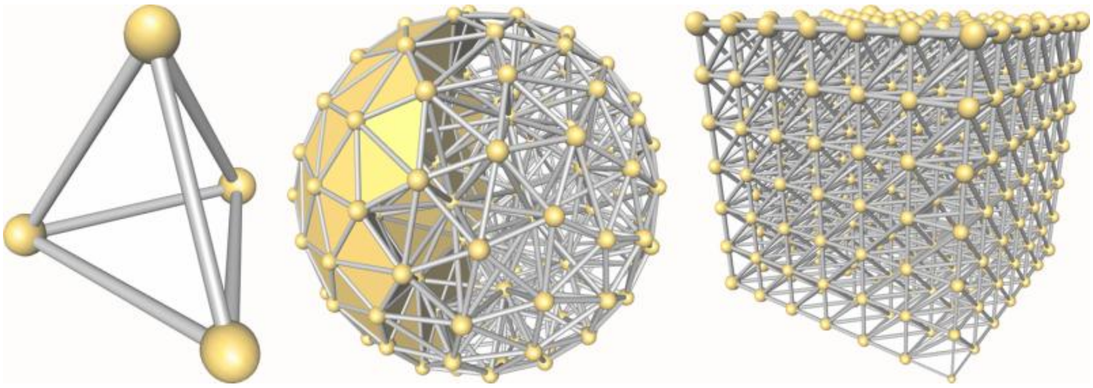


Figure 2.71 3D triangulation

There are many possible triangulations; the adopted one in this dissertation is the Delaunay triangulation. This triangulation was invented by Boris Delaunay in 1934. Delaunay is a particular triangulation that is uniquely determined from the point locations. It also has the following two geometric properties:

1. The circumcircle passing through a generic Delaunay triangle does not contain any other point.
   In the image number 15, the circumcircles are blue, the points are black and the triangles are black. The red straight lines are the Voronoi tesselation.

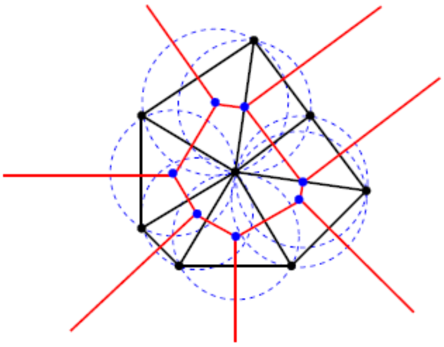


Figure 2.72 Triangulation and circumcircles

2. The triangles are as equilateral as possible: this fact implies that the minimum angle must be maximized.

When rotating a random side of the triangle, the minimum angle of the two adjacent triangles decreases. The two following images show a correct Delaunay triangulation (Figure 2.73a) and wrong one (Figure 2.73b).

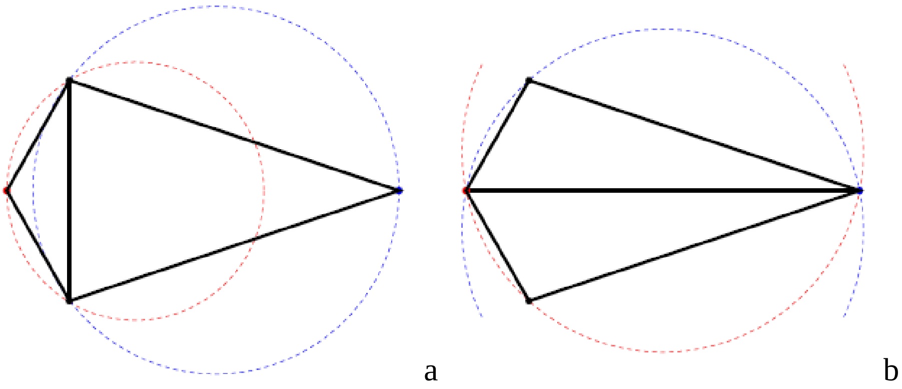


Figure 2.73 Correct and incorrect Delaunay triangulation

The advantages of this kind of interpolation are two:

- the input data belong to the interpolated field and
- the density of the interpolated points is the same of the input data.

The disadvantage is that the triangulation procedure is not robust. That is the reason why the input data must be checked before analyzing the data in terms of triangulation.

The purpose of this section of the dissertation is to better understand the Delaunay triangulation using examples. Figure 2.74 shows the 6 random points.

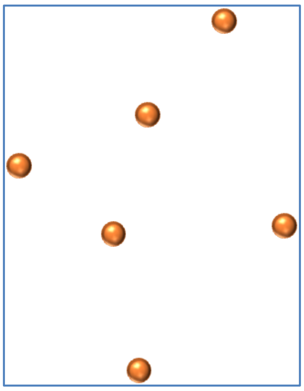


Figure 2.74 Input points

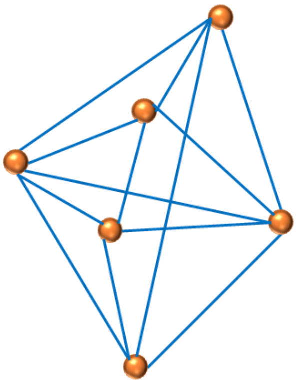


Figure 2.75 Random triangulation

The previous picture shows a wrong triangulation because:

- the triangle should be as equilateral as possible,
- a circumcircle should not contain any of the other points (it is shown in figure 2.56), and
- the smallest inner angle must be at its maximum breadth.

Figure 2.76 shows the correct Delaunay triangulation.

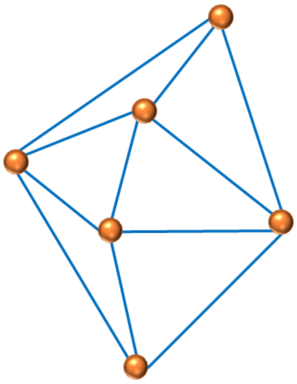


Figure 2.76 Delaunay triangulation

If the input point cloud does not have any holes, Delaunay triangulation perfectly works so no adaptations have to be done.

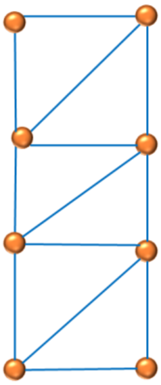


Figure 2.77 Correct Delaunay triangulation

If the input point cloud has a hole, it is not possible to directly apply Delaunay triangulation because some problems must be solved. In fact, if the body has a hole, the 3D model must have the same hole. Figure 2.78 shows the eight points in orange, the contours in dashed blue lines and the triangles in light blue.

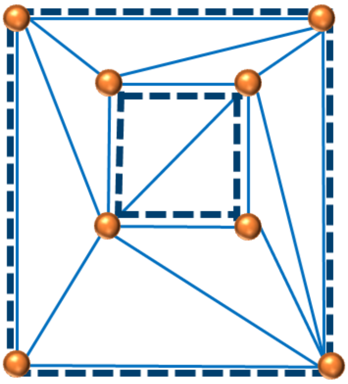


Figure 2.78 Input points (in orange), contours (in blue) and Delaunay triangulation (in light blue)

The software checks if the middle point of the side of the triangle is inside the closed path, on the perimeter or outside the closed path. If it is inside the two contours or on the perimeters, the software keeps it, otherwise the line is deleted. Figure 2.79 shows the middle point (in red) of the side P_1P_2. Because it is outside the perimeter, the side P_1P_2 is deleted.

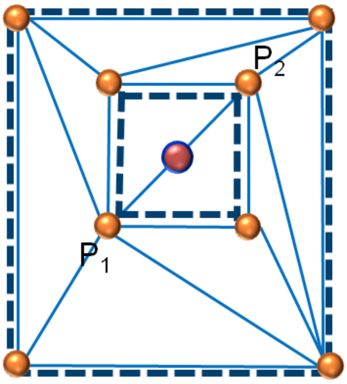


Figure 2.79 Delaunay triangulation, contours and the checking point

Figure 2.80 shows the final Delaunay triangulation.

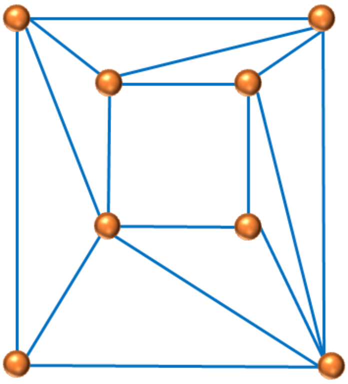


Figure 2.80 Correct Delaunay triangulation

If the input point cloud represents a concave body, Delaunay triangulation does not work so adaptations have to be done.

Figure 2.81 shows the ten input points in orange, the contour in dashed blue line and the Delaunay triangulation in light blue.

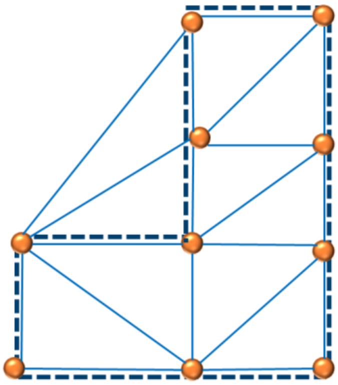


Figure 2.81 Input points (in orange), Delaunay triangulation (in light blue) and contour (in dashed blue)

The software checks if the middle point of the side of the triangle is inside the closed path, on the perimeter or outside the closed path.

If it is inside or on the perimeter, the software keeps it, otherwise the side is deleted. Figure 2.82 shows the middle point (in red) of the side P_1P_2 and the middle point (in red) of the side P_1P_3.

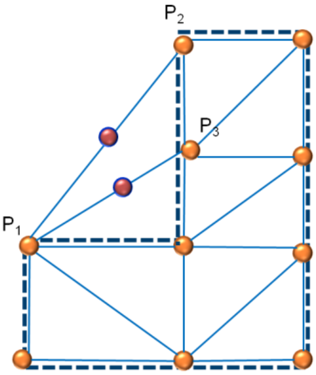


Figure 2.82 Delaunay triangulation, contour and the two control points

Because these two middle points (in red in the picture) are outside the perimeter, the sides P_1P_2 and P_1P_3 are deleted.
Figure 2.83 shows the final correct Delaunay triangulation.

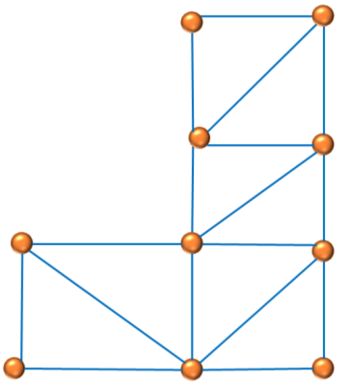


Figure 2.83 Correct Delaunay triangulation

There are several Delaunay triangulation applications. Delaunay triangulation is applied to the structure in two different directions: vertically to create a row surface and horizontally to create continuous models, one for each horizontal cluster.
Delaunay triangulation is applied to the infrastructure to calculate the area of the surface comprised of the space between the points connected using the relational matching.
Delaunay triangulation is applied to the aquifer to create a row surface for each cluster and to obtain the perimetration.
The following pictures are the results achieved from the Delaunay triangulation applied to the structure case study.
The one on the left is the triangulation obtained without any changes; the one on the right is the correct one. It is the correct because the horizontal cluster that generated the triangulation contains a hole so the triangulation must contain the same hole.

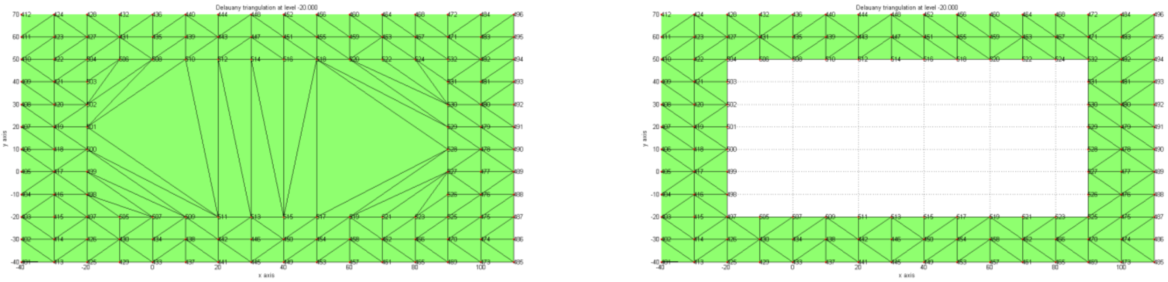


Figure 2.84 Wrong and correct triangulation at -20.000 m

The following pictures show two Delaunay triangulations. The one on the left is the triangulation obtained without any changes; the one on the right is the correct one.

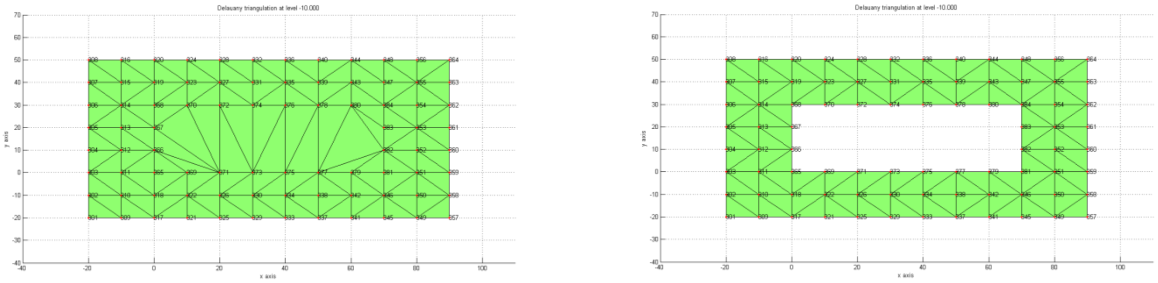


Figure 2.85 Wrong and correct triangulation at -10.000 m

The following picture shows the correct Delaunay triangulation of the horizontal cluster at 0 meter.

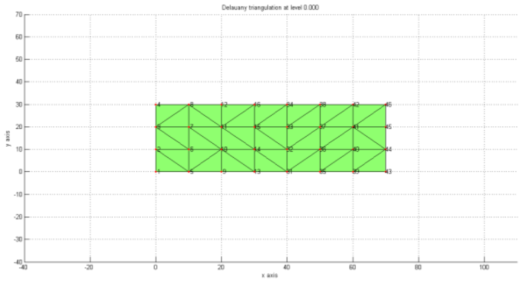


Figure 2.86 Delaunay triangulation at 0.000 m

The following pictures show two Delaunay triangulations. The one on the left is the triangulation obtained without any changes; the one on the right is correct. It is correct because the vertical cluster contains two horizontal clusters. There must be two different triangulations, one per each horizontal cluster.

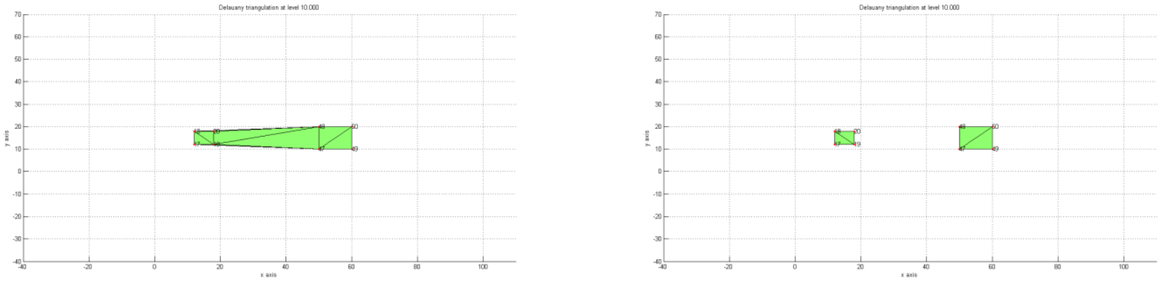


Figure 2.87 Wrong and correct Delaunay triangulation at 10.000 m

The following pictures show two Delaunay triangulations. The one on the left is the triangulation obtained directly without any changes; the one on the right is the correct one. The meaning of the two pictures is the same of the previous figure.

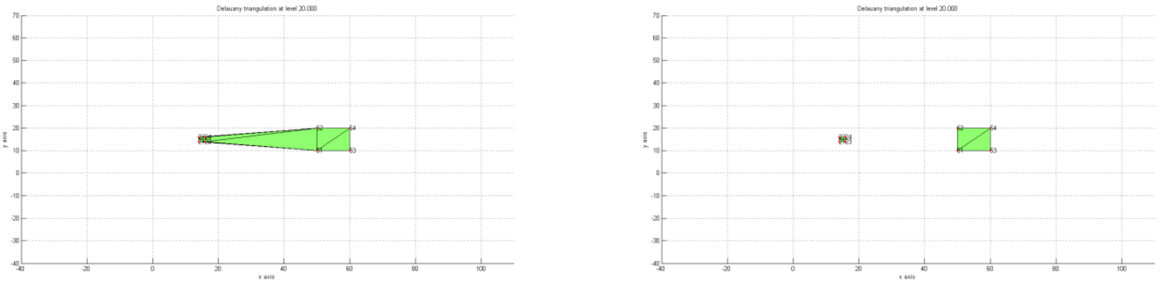


Figure 2.88 Wrong and correct Delaunay triangulation at 20.000 m

The following picture shows the correct Delaunay triangulation of the horizontal cluster at 30 meter.

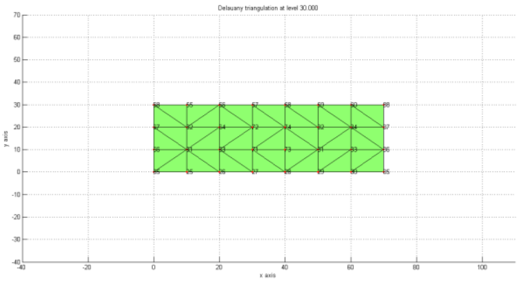


Figure 2.89 Delaunay triangulation at 30.000 m

The following pictures show two Delaunay triangulations. The one on the left is the triangulation obtained directly without any changes; the one on the right is the correct one. It is the correct one because the horizontal cluster that generated the triangulation contains a hole so the triangulation must contain the same hole.

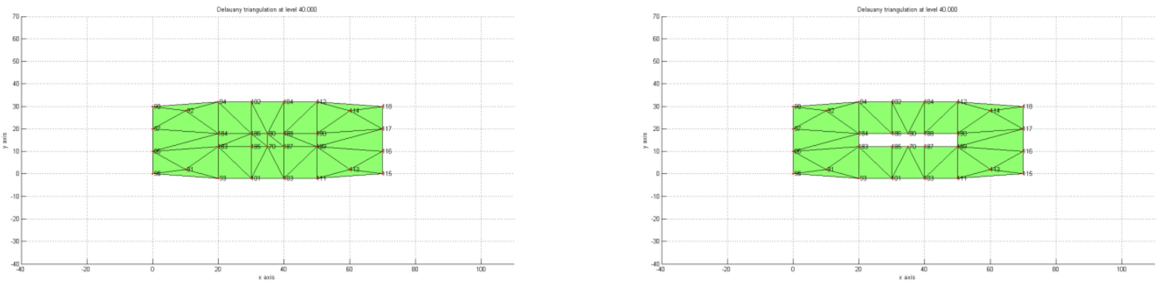


Figure 2.90 Wrong and correct Delaunay triangulation at 40.000 m

The following pictures show two Delaunay triangulations. The one on the left is the triangulation obtained directly without any changes; the one on the right is the correct one. The meaning of the two pictures is the same of the previous figure.

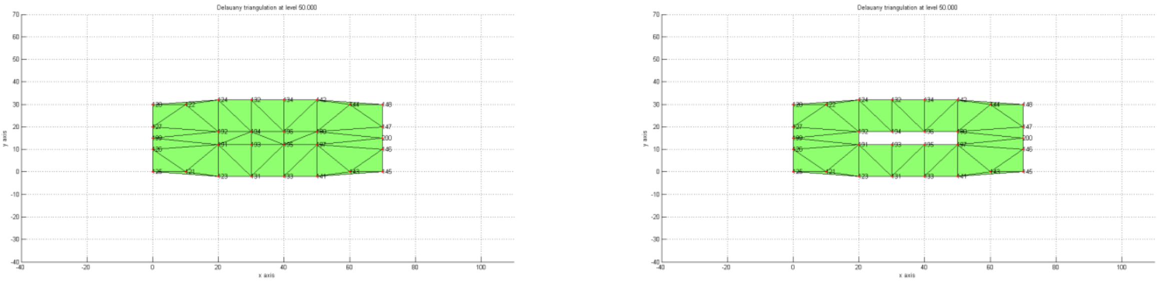


Figure 2.91 Wrong and correct Delaunay triangulation at 50.000 m

The following picture shows the correct Delaunay triangulation of the horizontal cluster at 60 meter.

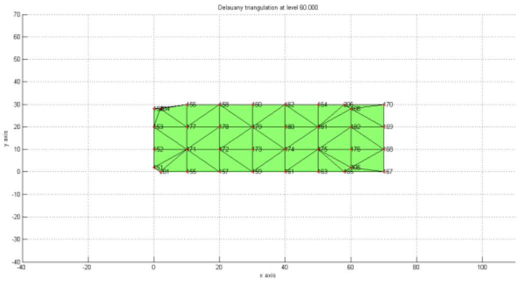


Figure 2.92 Delaunay triangulation at 60.000 m

The following picture shows the correct Delaunay triangulation of the horizontal cluster at 70 meter.

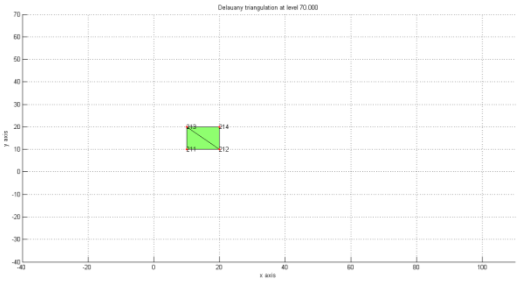


Figure 2.93 Delaunay triangulation at 70.000 m

The following picture shows the correct Delaunay triangulation of the horizontal cluster at 80 meter.

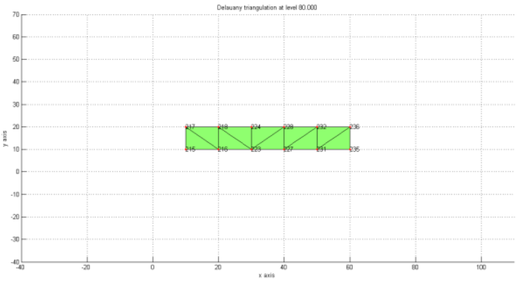


Figure 2.94 Delaunay triangulation at 80.000 m

The following picture shows the correct Delaunay triangulation of the horizontal cluster at 90.000 meter.

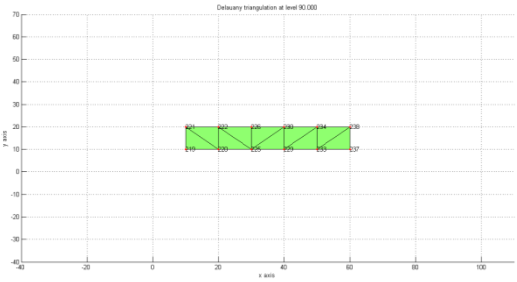


Figure 2.95 Delaunay triangulation at 90.000 m

The following picture shows the correct Delaunay triangulation of the horizontal cluster at 100.000 meter.

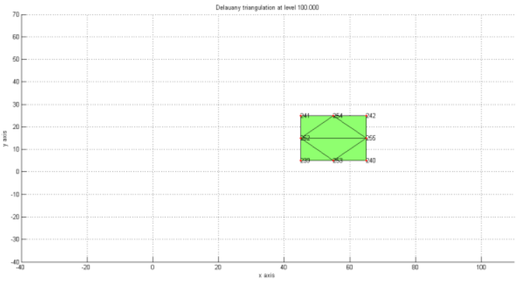


Figure 2.96 Delaunay triangulation at 100.000 m

The following picture shows the correct Delaunay triangulation of the horizontal cluster at 110.000 meter.

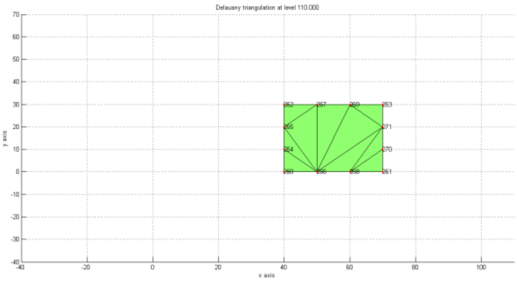


Figure 2.97 Delaunay triangulation at 110.000 m

The following picture shows the correct Delaunay triangulation of the horizontal cluster at 120.000 meter.

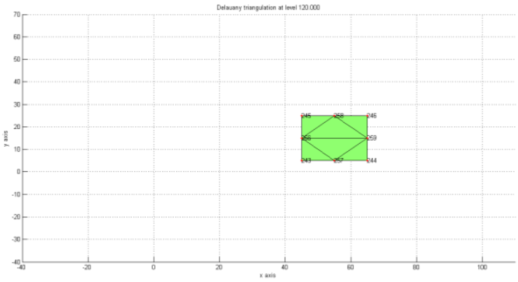


Figure 2.98 Delaunay triangulation at 120.000 m

The following picture shows the correct Delaunay triangulation of the horizontal cluster at 130.000 meter.

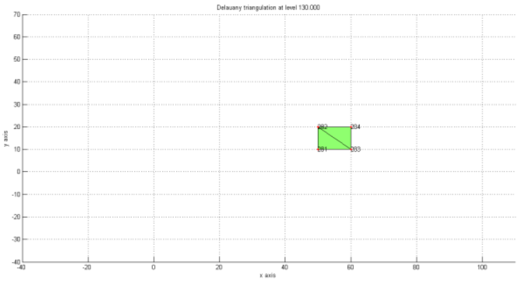


Figure 2.99 Delaunay triangulation at 130.000 m

The following picture shows the correct Delaunay triangulation of the horizontal cluster at 140.000 meter.

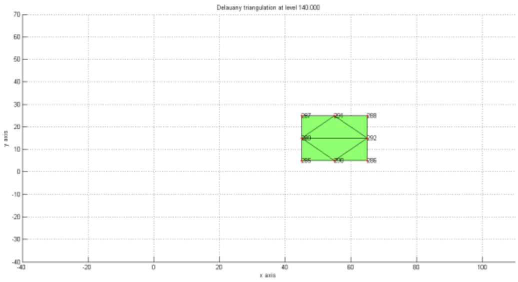


Figure 2.100 Delaunay triangulation at 140.000 m

The following picture shows the correct Delaunay triangulation of the horizontal cluster at 150.000 meter.

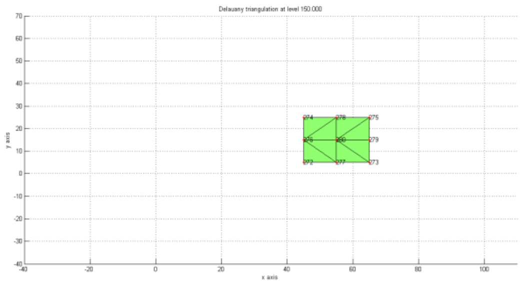


Figure 2.101 Delaunay triangulation at 140.000 m

As mentioned before, Delaunay triangulation has many applications. I will explain here the one I thought to apply if the case study is an infrastructure. Delaunay triangulation is applied after relational matching. This technique is explained in subchapter 5. As mentioned, the aim is to obtain the Brückner profiling automatically. To do so, the diagram of the areas is needed. Delaunay triangulation and stabilized Erone formula serve this purpose.

The following picture shows the points in orange, the relational matching in red and green, and the Delaunay triangulation in light blue.

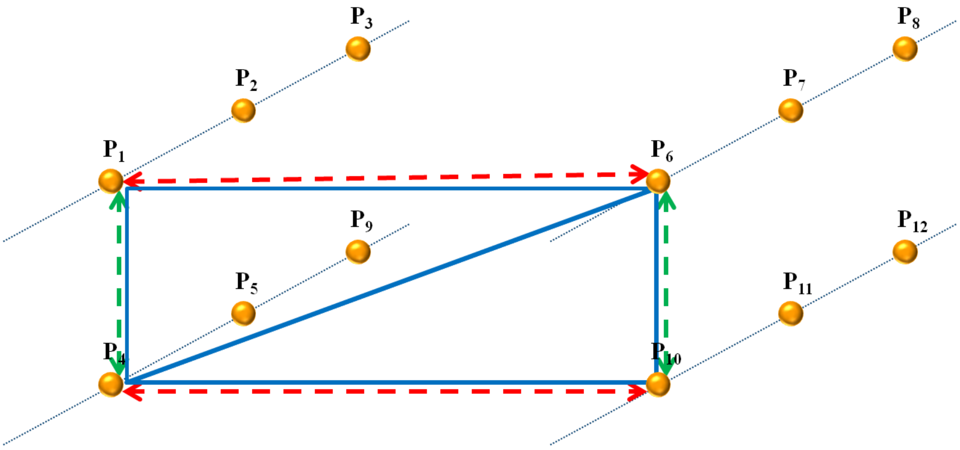


Figure 2.102 Delaunay triangulation applied after the relational matching

It is possible to apply Delaunay triangulation to every group of four points matched using the vertical and horizontal relational matching. The values of the area delimited by Delaunay triangulation are obtained using the stabilized Erone formula. This formula will be explained in subchapter 9.

As mentioned before, another application of the Delaunay triangulation is to generate the closed path if the point cloud represents a convex body. Figure 2.103 shows the eight input points in orange and their point IDs in black.

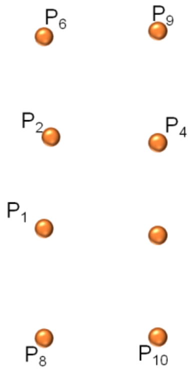


Figure 2.103 Input points

Figure 2.104 shows the eight input points in orange, their point IDs in black, and the Delaunay triangulation in light blue.

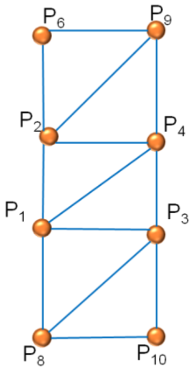


Figure 2.104 Delaunay triangulation of the input point cloud

Figure 2.105 shows the eight input points in orange, their point IDs in black, and the perimeter in dashed line. It is obtained from the Delaunay triangulation keeping only the sides not in common with other triangles.

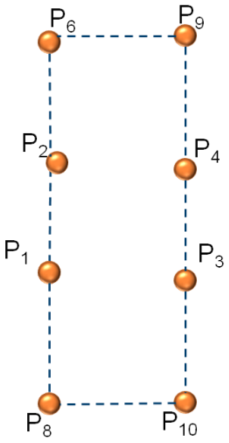


Figure 2.105 Delaunay triangulation of the input point cloud

2.7. INTERPOLATION

The goal is to obtain denser information, starting from the input points and the triangulation. Figure 2.106 shows the nine starting points in orange.

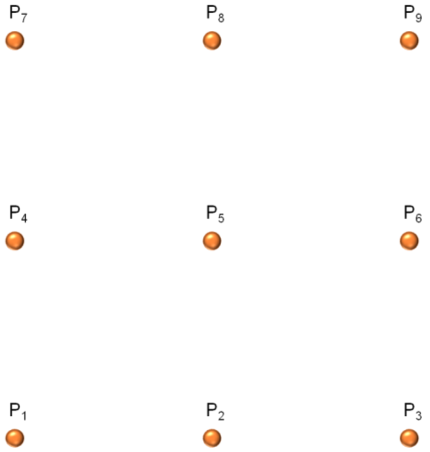


Figure 2.106 Input points

Figure 2.107 shows the nine starting points in orange and the Delaunay triangulation in blue.

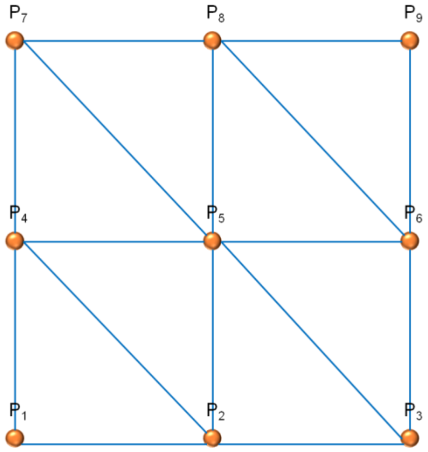


Figure 2.107 Delaunay triangulation

The aim at this point is to obtain the coordinates of new points. They are found using the parametric equation of a line. The lines used are the sides obtained by Delaunay triangulation.
If the coordinates of the input points are $P_0(x_0, y_0, z_0)$ and $P_1(x_1, y_1, z_1)$, the parametric equation of the line passing through those two points is the following:

$$x = x_0 + (x_1 - x_0) * t$$
$$y = y_0 + (y_1 - y_0) * t$$
$$z = z_0 + (z_1 - z_0) * t$$

If the parameter t is equal to 0, the coordinates found are the ones of the point P_0.

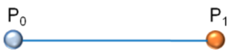


Figure 2.108 First new point

If the parameter t is equal to 1, the coordinates found are the ones of the point P_1.

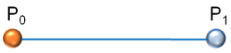


Figure 2.109 Second new point

If the parameter t is equal to 0.5, the coordinates found are the ones of the middle point between P_0 and P_1.

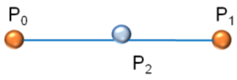


Figure 2.110 Third new point

Using this procedure, it is possible to obtain new points. The results showed in figure 2.111 are obtained:

- t equal to 0,
- t equal to 0.33,
- t equal to 0.67, and
- t equal to 1.

The following image shows the input points in orange, the Delaunay triangulation in blue and the new points in light blue.

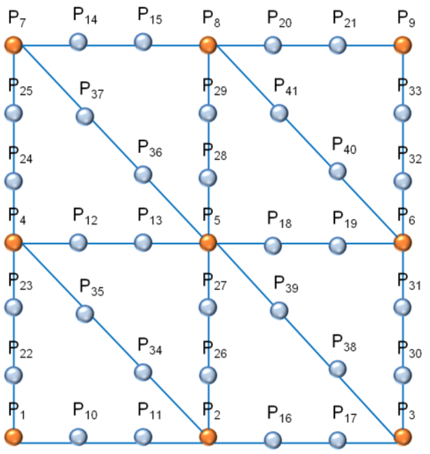


Figure 2.111 New points

The software provides the new point IDs and the three coordinates.

2.8. BRÜCKNER PROFILE ANALYSIS

The Brückner profile analysis is highly adopted in highway engineering and it consists in the calculation of cut and fill volumes of soil during a road construction project. It is also applied in an airport construction. In particular, the Brückner profile can inform engineers about soil handling quantities to be provided or removed for each road section. Therefore, analyzing soil quantities among the whole longitudinal road or airport profile allows the optimization of materials to be handled in order to minimize costs and hauling distances.

The Brückner diagram can be obtained by integrating the area diagram for two consecutive road or airport sections. So far these sections are achieved manually.

I will briefly explain how the diagram of the area is obtained and how it is possible to understand. Two consecutive sections are displayed in figure 2. Section 1 is characterized by an area equal to S1 and section 2 is characterized by an area equal to S2, and the two sections are distance D.

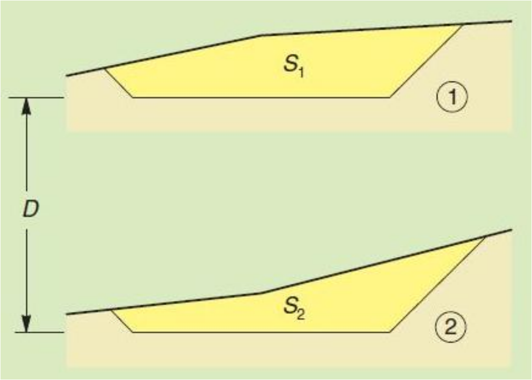


Figure 2.112 Two starting sections

It is possible to build the diagram of the area in this way. The distance is represented by a segment in an appropriate scale in the x-axis; the two areas are represented by two vertical segments (AA’ and BB’) in an appropriate scale in the y-axis.

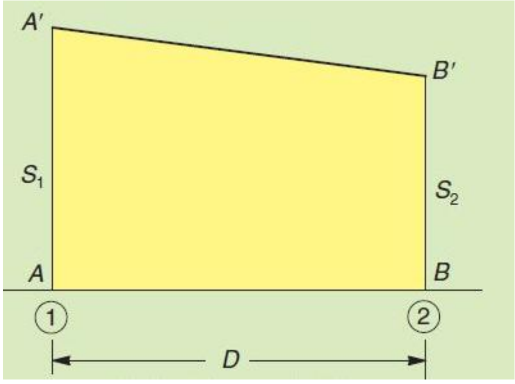


Figure 2.113 Two starting areas

Two different consecutive sections are displayed in figure 2. Section 1 (the cut section) is characterized by an area equal to S1 and section 2 (the fill section) is characterized by an area equal to S2, and the two sections are distance D. The area is positive till distance d_S, and then it is negative at distance d_R.

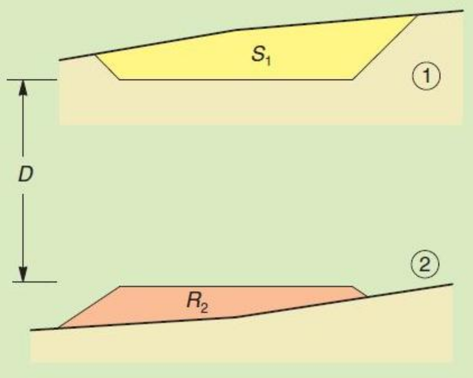


Figure 2.114 Other two starting areas

The two distances (d_S and d_R) are represented by two consecutive segments in an appropriate scale in the x-axis, the two areas are represented two vertical segments (AA' and BB') in an appropriate scale in the y-axis.

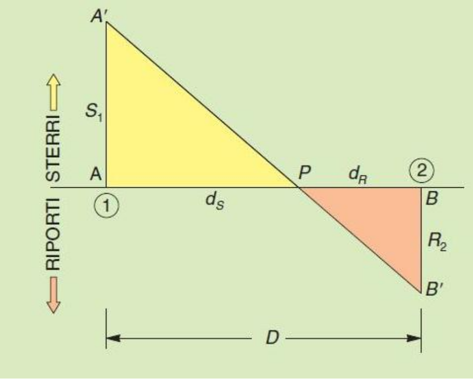


Figure 2.115 Other two starting areas

The area comprised between two vertical segments and the distance between them is proportional to volume of the soil contained in between that section. If the area in the diagram is positive, that zone is classified as a cut. On the opposite, if the area in the diagram is negative, that zone is classified as a fill. The following picture shows a full example of the diagram of the areas.

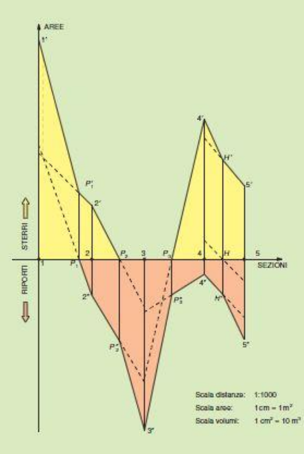


Figure 2.116 Diagram of the areas

The goal is to understand where it is possible to move the soil volume. It is necessary to overturn the positive part of the diagram of the area on the negative part and then to overturn the negative part of the diagram of the area on the positive part.
The area in common is highlighted in blue in the following picture.

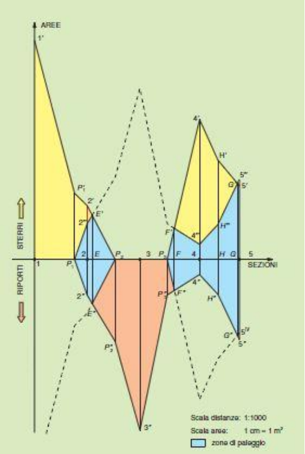


Figure 2.116 The diagram of the areas and the two overturns

The area highlighted in light blue is equal to the double volume of the soil that can be moved from the cut sections to the fill sections. A new diagram of the areas is therefore obtained, deleting the light blue parts.

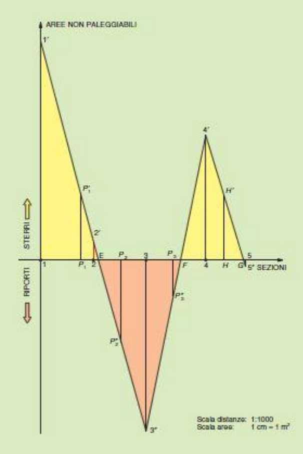


Figure 2.117 The new diagram of the areas

The Brückner profiling is obtained from the last diagram of the area through graphic integrations.

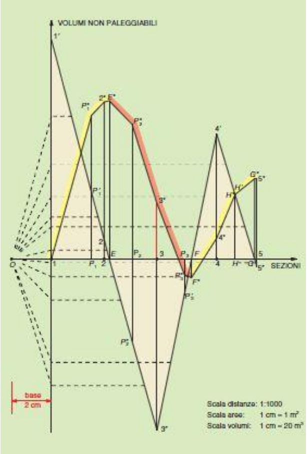


Figure 2.118 the Brückner profiling

2.9. STABILIZED ERONE

Erone from Alessandria discovered the formula showing that it is possible to find the area of a triangle starting from the length of its sides. Al-Burini claimed the same formula was invented by Archimede. The Erone formula says that the area of a triangle whose sides are length a, b, and c is given by the following formula:

$$A = \frac{1}{4} * \sqrt{p * (p-a) * (p-b) * (p-c)}$$

with p the semi-perimeter $p = \frac{a+b+c}{2}$.
The previous formula is not stable if one of the angles is narrow. A stable alternative requires the preparation of the sides so that $a \geq b \geq c$ and the use of the following formula:

$$A = \frac{1}{4} * \sqrt{\big(a + (b + c)\big) * \big(c - (a + b)\big) * \big(c + (a - b)\big) * \big(a + (b - c)\big)}$$

The brackets are necessary to obtain the required stability.
The Erone formula is used after the two following procedures:

- Relational matching and
- Delaunay triangulation.

With the former it is possible to obtain the connection in the horizontal and vertical direction. With the latter it is possible to join those points connected with the relational matching. After those steps, it is possible to evaluate the area of the triangles generated by the Delaunay triangulation.
The following picture displays the points in orange, the relational matching in light blue, the Delaunay triangulation in orange. The Erone formula is applied last.

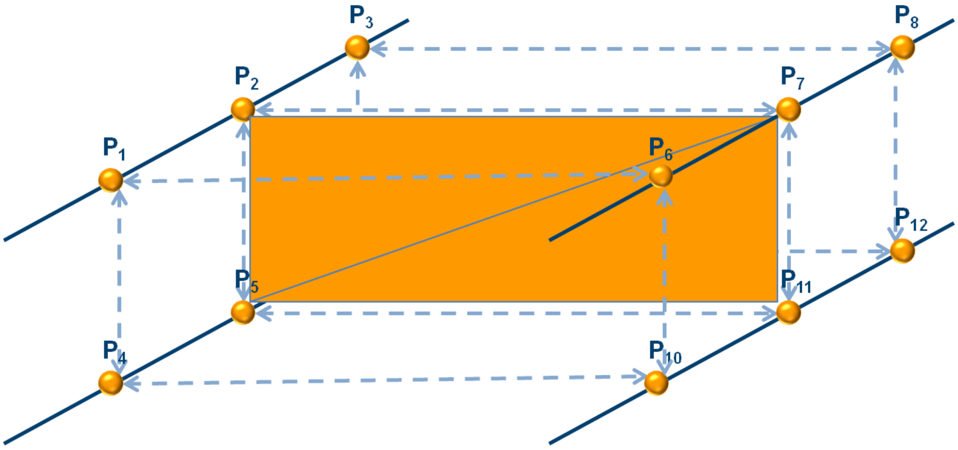


Figure 2.119 Relational matching and Delaunay triangulation

It is possible to obtain the area diagram automatically using these three techniques in sequence. This technique is therefore applied if the case study is an infrastructure.

CHAPTER 3

LiDAR data

The following is a brief introduction to this type of data.

Electromagnetic energy is continuously emitted from any object whose temperature is above absolute zero (-273˚C, -459.4 °F). Electromagnetic energy can be absorbed, scattered, reflected, or transmitted. The Sun represents the initial source of most of the electromagnetic energy recorded on Earth by remote sensing systems.

Light is reflected from features, and the electromagnetic radiation can be collected by a device calibrated to record the amount of energy in specific wavelengths. The acronym "laser" stands for "light amplification by stimulated emission of radiation." A laser is a device which generates a stream of high energy particles within an extremely narrow range of wavelengths. Lasers produce a coherent light source. A laser light source forms the basis for a LiDAR system. The term "LiDAR" is an acronym for "light detection and ranging." The wavelength chosen for most airborne topographic mapping lasers is 1064 nm, 4.189*10^{-5} inches, 3.491*10^{-6} feet.

There are two kinds of sensors: passive and active. The former ones record the amount of ambient energy reflected or emitted from features of interest; the latter, such as LiDAR, generate and transmit energy in specific wavelengths and measure the amount of that energy reflected back from target surfaces. Since these sensors do not require sunlight, they can be used either during the day or at night. Longer wavelengths, such as microwaves, can travel uninterrupted through clouds, while shorter wavelengths cannot.

The essential measurement made by a LiDAR sensor is the time that elapses from the moment the pulse is emitted until it returns after being reflected by the target surface.

Because the laser pulse travels at the speed of light, a known constant, time can be directly converted to distance, by the following equation:

$$r = \frac{\Delta t * c}{2}$$

Here r is the distance from the sensor to the target and back, Δt the elapsed time, and c the speed of light. The distance is often referred to as the range.

Therefore using LiDAR, it is possible to measure the time it takes an emitted pulse of light to travel to an object, be reflected off the object, and travel back to the sensor and precisely locate the position of the sensor.

If the laser pulse strikes a solid object the incident energy is all reflected and only one return is recorded. Very dense vegetation may also reflect all of the incoming energy at the same time, so that only one return will be recorded. But, if the vegetation canopy allows for any of the laser light to continue traveling forward between individual leaves, stems, and trunks, additional returns may be recorded from within the canopy, and possibly even from ground beneath the canopy.

It is technically possible to record the entire waveform of the returning laser pulse. Doing these, the analysis of vegetation density, mapping live versus dead vegetation, forest fuels analysis, and wildlife habitat mapping is possible.

Very early LiDAR systems actually recorded only one discrete return, either the first peak or the final peak in the reflected wave. By 2000, commercial systems became capable of measuring multiple (3 -5) returns per pulse. These multiple returns can be analyzed and classified to produce information about objects above the ground as well as the bare ground surface. Figure 3.2 shows the possible 5 multiple returns from a survey mapping.

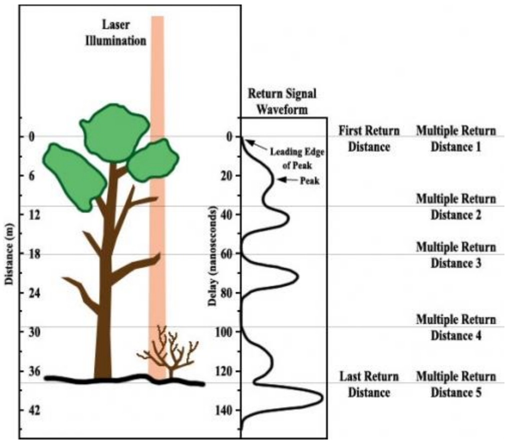


Figure 3.1. Multiple returns

The number of first and second returns is typically quite large compared to additional or last of many returns. This is because the laser pulse energy is reflected by the canopy and sub-canopy forest structure, the amount of laser light available to record returns decreases with the depth as it goes through the vegetation .

The processed LiDAR data, before classification, is essentially a cloud of 3D points composed of "single returns" or "multiple returns" from laser pulses.

LAS points can be classified into a number of categories. Predefined classification schemes are defined by American Society for Photogrammetry and Remote Sensing (ASPRS). ASPRS Standard LIDAR Point Classes are the following ones:

Classification Value	Meaning
0	Created, never classified
1	Unclassified

2	Ground
3	Low Vegetation
4	Medium Vegetation
5	High Vegetation
6	Building
7	Low Point (noise)
8	Model Key-point (mass point)
9	Water
10	Reserved for ASPRS Definition
11	Reserved for ASPRS Definition
12	Overlap points
13 - 31	Reserved for ASPRS Definition

Table 3.1 LiDAR data classification

The study cases that use LiDAR data are Levy County and Marion County in Floirda, U.S.A. The unprocessed LiDAR data are classified into these three classes:

Classification Value	Meaning
1	Unclassified
2	Ground
9	Water

Table 3.2 Input point cloud LiDAR data classification

CHAPTER 4

Shrub

A shrub or bush is a woody plant characterized by several stems arising from the base and height usually under 6 m (19.658 ft) tall. Shrubs generally have dense foliage and many small leafy branches growing close together.
Figure 3.1 provides an example of this plant.

Figure 4.1 Shrub

For shrubs less than 2m (6.56 ft) high the following structural forms are categorized:

- very sparse foliage cover (<10%) — low open shrubland,
- sparse foliage cover (10–30%) — low shrubland,
- mid-dense foliage cover (30–70%) — open-heath or mid-dense low shrubland—(North America), and
- dense foliage cover (70–100%) — closed-heath or closed low shrubland—(North America).

For shrubs 2 m (6.56 ft) – 6 m (19.658 ft) high the following structural forms are categorized:

- very sparse foliage cover (<10%) — tall open shrubland,
- sparse foliage cover (10–30%) — tall shrubland,
- mid-dense foliage cover (30–70%) — open-scrub, and
- dense foliage cover (70–100%) — closed-scrub.

Since some of the evergreen shrubs can often grow in dense thickets and are nearly impossible to traverse, information on the location of particularly dense stands is useful for field scientists, rescue workers and resource.

CHAPTER 5

How to rescue people

The topics related to this subject are covered in everyday life, newspapers, and papers and are basically three. The first one is how to rescue people in general and weak people in particular, the second one is how to use dogs and their help during rescue operations, and the last topic is focused on who should pay the rescuers.

5.1. LOST PEOPLE

The first covered topic relates to people: the weak ones are the ones with mental health problems, the elderly, despondent subject, those with Alzheimer's, children, and young people. For example, people with Alzheimer Disease become confused and get lost, often ending up in a secluded spot hidden by brush or other covers. Many victims are elderly and the aim of the rescuers is to find the lost person as quickly as possible or to determine that the person is not in the area being searched. Researches have shown that if a lost person with Alzheimer disease who is not found within 12 hours, there is a 50% chance that he/she will be found injured or dead from hypothermia, dehydration, or drowning.
To better understand effective search, the behavior of lost persons must be understood. A summary of studies on their behavior includes:

- if a lost person finds a trail, she/he might get on it and run - convinced she/he is on the way back, when in fact she/he might be running FROM it,
- rarely will a lost person reverse their direction on a trail,
- many people ignore trails and follow their own logic - traveling in a straight line. They figure they will come to a road or highway - not expecting the cliff or impassable river that ultimately confronts them,

- some lost people will climb/hike to the top of the closest hill to get a better view, only to find that the trees atop that hill obstruct any view [Charley Shimanski, Search and Rescue for outdoor leaders],
- the majority of lost people will travel downhill and/or downstream,
- those who travel downstream will likely end up in a swamp or impassable confluence long before they reach civilization,
- many lost people will travel at night - even without a flashlight,
- most lost persons will stay on a trail if they're not absolutely sure of the right direction, and
- lost people will rarely move around randomly - they usually move with conviction and hope that they are heading in the right direction.

Another problem that should be taken into account is the following: what happens if the subject is semi-ambulatory?

If the subject is ambulatory, evacuation might be an assisted walkout. But in the more difficult situations, evacuation might only be possible by using a rescue litter. And if the terrain is real nasty, a technical rescue system may be required to lower the litter to more benign terrain. A trail carry usually involves 4 to 6 litter bearers that walk on each side of the litter and carry it down an established trail. It can be a challenge because most trails are not wide enough for a litter and attendants, that are as wide as 3 people.

5.2. RESCUE DOGS

The second topic relates to rescue dogs. Why that kind of dogs is used in the emergency situations? It has been documented that one well trained air scenting dog team can be effective as 50 - 100 ground searchers. This is because a dog's sense of smell is millions of times better than a person's sense of smell.

Scent consists of tiny bits of skin cells, referred to as "rafts" or "scurf", that carry a person's odor because they are covered with sweat and acted on by bacteria. Because the human body sheds about 50 million cells each minute, there rafts fall from the body like a shower on microscopic snowflakes. They can percolate through air, water, snow, mud and even porous concrete. With their noses tuned for biological odors and pheromones, dogs easily detect these rafts and can follow them to the person who shed them. Detecting the traces of the person who passed is just the first part of the dog's task. He also must determine which way the person went. Earlier footsteps are a few seconds older and just a fraction weaker in the intensity of the scent that they give off. This gives the dog the clue that he needs to follow in the correct direction, toward the stronger scent. However, the dog needs distinct patches of scent to do this correctly. As a trail grows older, the scents fade, which makes determining its direction much more difficult. Warm days with sunshine will weaken a scent quickly. The ultraviolet rays destroy some of the odor-bearing chemicals, eliminating much of the scent.

Before training a dog to find people, it must be thoroughly understood what the dogs are responding to and the effect that wind and terrain have on scent transport.

Handlers in even the early stages of training must understand the effect that terrain has on scent behavior. Open fields, light shrub, heavy shrub, woods and drainages are the different possible terrain.

Light shrub is defined as space that includes open or wooden areas with some brush or small woodpiles. Light shrub problems should not pose any real difficulty for the dog - the brush should not be so thick that it blocks or drastically changes scent flow.

On the contrary, heavy brush may be found in thick woods or unmowed fields. A heavy brush area may include brier patches and large woodpiles. Still, hot days combined with heavy shrub can produce extremely difficult searching conditions because the scent will remain near the victim. Detection and ranging distances will be necessary for accurate coverage. The question is: how is it possible to classify the shrub layer in the LiDAR point cloud into light and heavy classes?

Another aspect that should be taken into account is how to planning the search. The operational leader should study all terrain features on the map that may affect the victim's travel route. Are there major barriers that would either be impassable or so obvious that the person would follow them (major roads, lakes, cliffs)? How is it possible to detect them starting from the input data?

Once the person is found, the dog has to go tell the handler (this is called an alert). There are two different alerts. The former one involves coming back to the handler. The handler then has the dog show them where the missing person is. But there is a problem: in the case that dog uses the first type of handler, the dog comes back to the lost person going straight to him. What happens if the dog attempts to pass through shrubs where, on the contrary, human beings are unable to cross?

In the second case of handler, the dogs is staying at the missing person's side and barking there, to bring the handler in to the missing person that way. In this case, how can the rescuer find the suitable path to reach the person?

5.3. RESCUE COSTS

The third topic deals with the training, gear and rescuers costs. Usually, the people who are getting rescued do not pay anything, but three aspects should be taken into account:

1. From 1992 to 2007, the U.S. National Park Service spent $58 million on search and rescue efforts. Volunteers give their time for free, but training and gear cost $25,000 a year.
2. When fire departments and military units are part of rescue efforts, they often have hours to log towards rescues. A real live rescue helps them meet their quota.
3. Rescue groups worry that people will delay calling for help if they fear they may be charged for it, and thus likely worsen a bad situation, including increased danger to the rescuers.

The best solution of this problem is being financially proactive. For example, Colorado collects a small portion of the money from state recreational fees to put into a fund that is earmarked for search and rescue. In Alaska, people who are mountain climbing up Mount McKinley pay $200 for the privilege. Another solution is a call asking for a donation to the county after a rescue.

5.4. ADDITIONAL INFORMATION

In addition to all these comments, three things must be taken into accounts.

The first one is that GPS tracking is a life saving device. GPS tracking is part of cellular phones. During periods of disasters, people can use this system to keep track to victims. If their mobile phone is active and is on GPS, their location can be easily tracked by rescuers.

The second one is the feeling of search teams returning from an assignment. They felt not very successful because the brush may be very dense. For this reason, it is helpful to classify the understory vegetation data set according to the density.

The third ones are practical hints in the unlucky event that somebody is lost in the woods and these are:

- Stay Together; DO NOT separate if you are with a friend or a pet.
- Stay in one place or area. DO NOT WANDER!
- Keep warm.
- Find a cozy waiting place, not a hiding place.
- Put out something bright so searchers can see it.
- Look bigger for searchers.
- Do not lie on the bare ground.
- Do not eat anything you are not sure of.
- Stay away from large rivers and lakes.

PART II

CHAPTER 6

The first case study:
the Central Television Tower Headquarters, in Beijing, China

This chapter deals with the reconstruction of a structure, starting from a point cloud. The test example is the China Central Television Tower Headquarters. It is a 234 m (768 ft), 44-story skyscraper on East Third Ring Road, Guanghua Road in the Beijing Central Business District (CBD) [Wikipedia].
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 the Office for Metropolitan Architecture (OMA) in August 2008 [Architecture now!]. This structure is represented in figure 6.1.

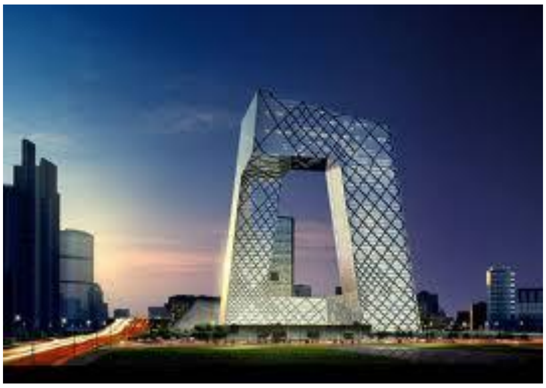


Figure 6.1 China Central Television Tower Headquarters

The input information is discrete, while the desired output is a continuous and consistent 3D model. The structure is composed of sowns and vertical or almost vertical walls.

As explained in chapter 2, section 4, a sown is the representation of a horizontal plane formed by dense points. A chain is a planar path modeled by rare points. Figure 6.2 shows a chain with dots in blue and a sown with dots are in red.

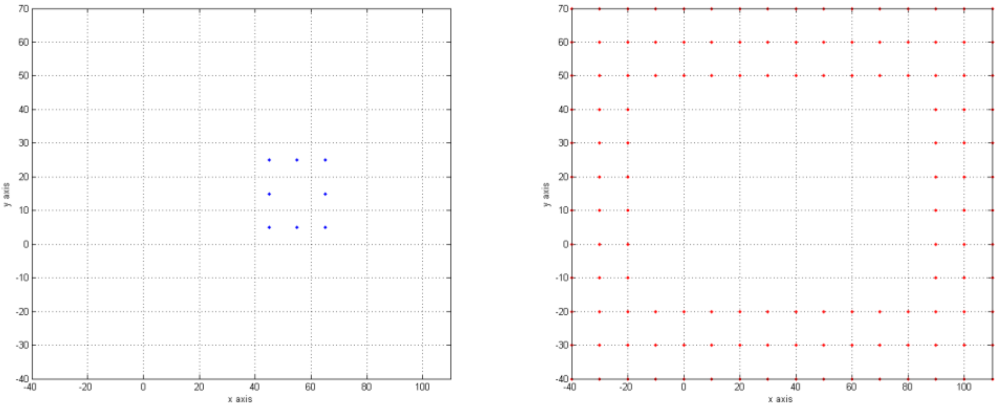


Figure 6.2 Chain (left) and sown (right)

The shape of this building is non-stellar, concave and multi–connected. If the shape is non-stellar, to reach all the points, it is necessary to go outside the structure. This fact is illustrated in figure 6.3. It is impossible to connect the points P_1 and P_2 without going outside the boundaries of the structure. Points P_1 and P_2 are in blue and the connecting line in orange.

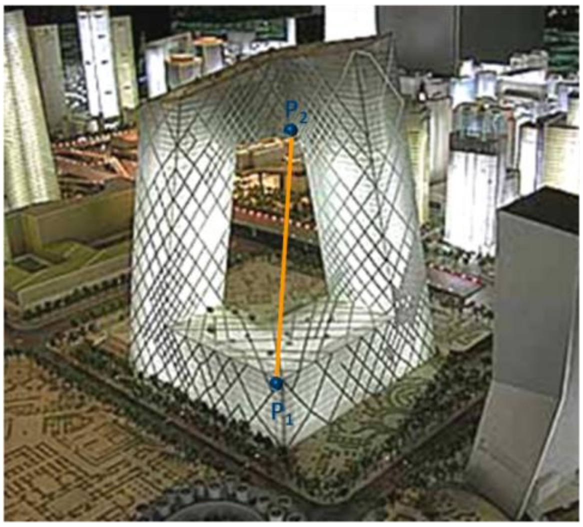


Figure 6.3 Visualization of the non-stellar shape

If the shape is multi-connected, it means there is a hole. The hole is highlighted in the picture 5.4 using an orange dashed line.

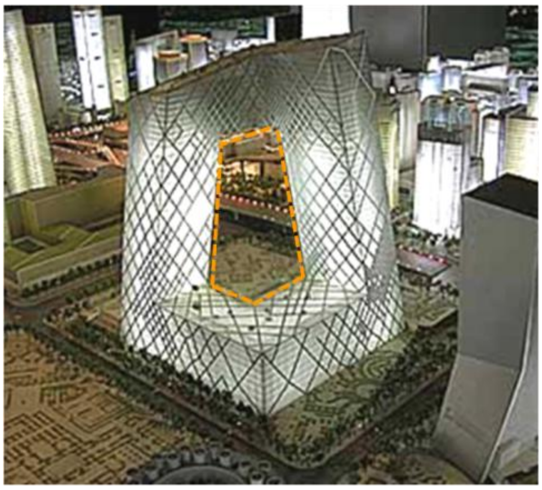


Figure 6.4 Visualization of the multi-connected shape

The China Central Television Tower Headquarters point cloud is modelled by 3047 points, acquired from pictures available on the web and in architectural literature. Each point is defined only by its three orthogonal Cartesian coordinates, without any other distinction. An extraction of the data set is shown in the following table.

ID	X	Y	Z
253	0.000	120.000	-20.000
273	200.000	120.000	-20.000
274	0.000	130.000	-20.000
294	200.000	130.000	-20.000
295	0.000	140.000	-20.000
...	...	...	...

Table 6.1 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 6.5 represents the reference system adopted and the scale.

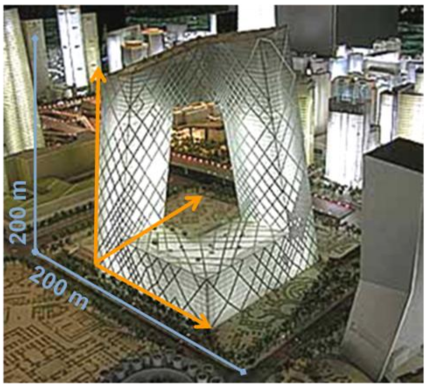


Figure 6.5 Reference system adopted

The results will be given in the same reference system in which the coordinates are provided. The input file needed is "CCTV.txt" and the 3D output plot given is "3D point cloud.tif". This first plot is shown in figure 6.6.

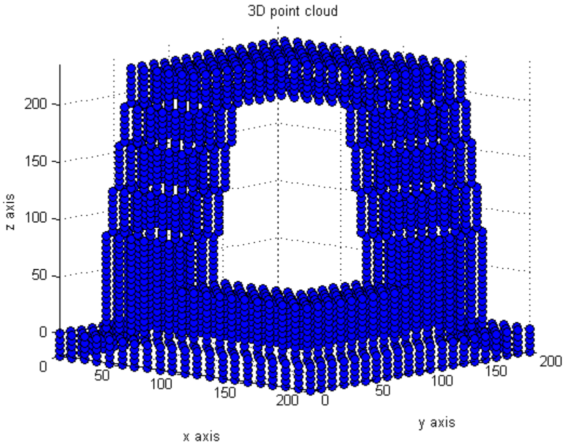


Figure 6.6 3D point cloud

The 3D reconstruction of a structure uses a sequence of procedures, because near impossible to use a single algorithm. [Forcella, Mussio. ISPRS, XXXVIII]
The integrated techniques and the procedures used are the following:

1. check the input data set,
2. the "voxel" procedure, a method based on octree encoding, is used,
3. cluster analysis,
4. extract features from the data set,
5. relational matching,
6. Delaunay triangulation.

The software used is Fortran® and Matlab®. The following subchapters explain in detail the different procedures.

6.1. OUTLIER DETECTION

The first procedure is devoted to checking the data set and to detecting the minimums and the maximums of X, Y and Z coordinates.
The input file needed is "CCTV.dat", the output file is "CCTVRev.dat", and the 3D output plot provided is "CCTV.tif".
The input file is "CCTV.dat" contains a list with the:

- point ID,
- X-coordinate,
- Y-coordinate, and
- Z-coordinate.

See table 6.1 for an example.
The data set must be checked because different points could have the same coordinates, different points in different locations could have the same ID number or points outside of the current layer may exist. If one of these three cases occurs, a list with the number ID, coordinate X, Y and Z is provided. If there are no duplicate points, it is shown. The output file “CCTVRev.dat” contains the checked data which will be used subsequently. In this specific case there are no duplicate points or points from another layer so “CCTV.dat” is equals to “CCTVRev.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.

Therefore 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, and
- +1.000 in Z axis.

and the adopted rotations are:

- 0 in X axis,
- 0 in Y axis, and
- 0 in Z axis.

The estimation of the three values needed (the height interval 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 the “contours” procedure are:

- height step: 8.000,
- planimetrical distance: 15.000, and
- planimetrical tolerance between points at different levels: 4.000.

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 m,
- X maximum: 200.000 m,

- Y minimum: 0.000 m,
- Y maximum: 200.000 m,
- Z minimum: -20.000 m, and
- Z maximum: 235.000 m.

This step is completed so it is now possible to move on to the following procedure.

6.2. VOXEL

As mentioned in chapter 2, section 2.
The goal of this procedure is to attribute points to their voxel and to estimate the three values needed afterwards. The input file needed is “CCTVRev.dat”, the output file given is “Voxel.dat”, and the 3D output plot given is “Voxel.tif”.
For each order of voxel, a list provides:

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

The “voxel” procedure continues up to the order in which the voxels are all empty.
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.
In this specific case, the results are:

- voxel 1° has no point,
- voxel 2° has no point,
- voxel 3° has 1 point,
- voxel 4° has 167 points,
- voxel 5° has 1,077 points, and
- voxel 6° has 1,802 points.

Figure 6.7 shows the voxel distribution. The meaning of the different colors are as follows:

- cyan is used for the points classified as belonging to voxel 1°,
- black is used for the points classified as belonging to voxel 2°,
- yellow is used for the points classified as belonging to voxel 3°,
- blue is used for the points classified as belonging to voxel 4°,
- red is used for the points classified as belonging to voxel 5°, and
- green is used for the points classified as belonging to voxel 6°.

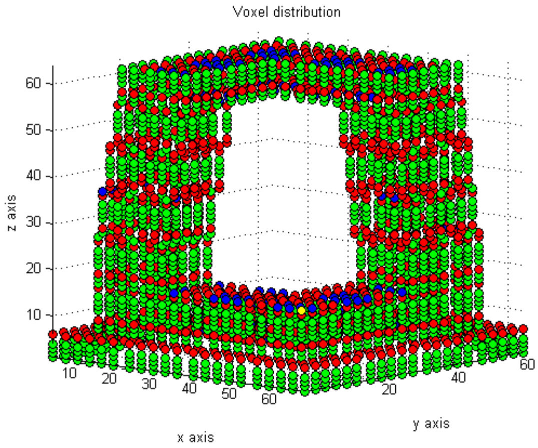


Figure 6.7 Voxel distribution

The “voxel” procedure is also used to the estimate the three values needed afterward.

- The first parameter defines vertical scanning and divides horizontal surfaces (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 vertical between points belonging to consecutive levels; this parameter is also used to distinguish sowns from chains.

In CCTV reconstruction, the parameters produced are:

- height interval: 7.969 m,
- planimetrical distance: 17.678 m, and
- planimetrical tolerance between points at different levels: 4.000 m.

This procedure is completed so it is possible to move to the following step.

6.3. CLUSTER ANALYSIS

The third procedure is devoted to analyzing the data in terms of cluster analysis. The input file is “CCTVRev.dat” and the output file is “CCTVCluster.dat”. The first parameter, estimated using the “voxel” procedure, is necessary to do vertical clustering.
For each level, the information is listed in the following way:

- level ID,
- number of points belonging to that level,
- level,
- point IDs at that level, and
- number of vertical clusters.

The extract of the vertical clusters list:

ID	Number of points	Height (m)	Point IDs			
1	80	-20.000	1	2	3	4
			5	6	7	8
			9	10	11	12
			…			
			438	439	440	441
2	80	-15.000	442	443	444	445
			446	447	448	449
			…			
			…			
			484	500	501	502
…	…	…	…			
11	64	30.000	2445	2446	2447	2448
			…	…	…	…
			2505	2506	2507	2508
…	…	…	…			
52	153	235.000	4338	4339	4340	4341
			…	…	…	…
			4493	4494	4495	4496

Table 6.2 Vertical clusters, table

In this case, there are 52 vertical clusters so one 3D plot and 52 3D plots are provided to the user, one for each vertical cluster.

The following picture (6.8) shows the 3D output plot named “3DVertClusters.tif”.

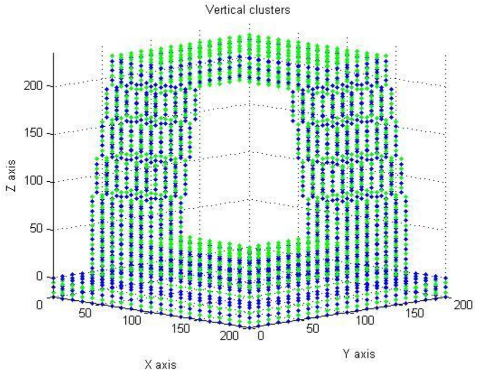


Figure 6.8 Vertical clusters

Here only two 2D plots are attached:

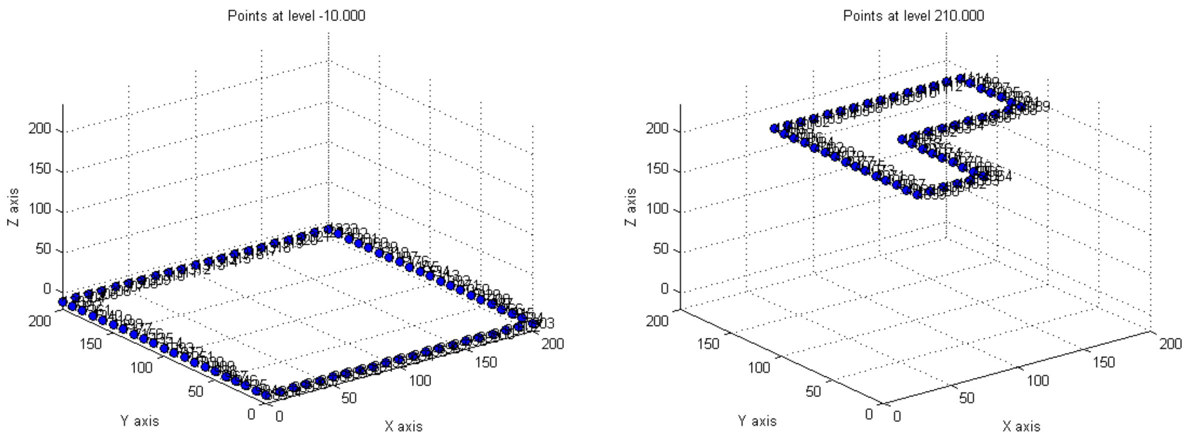


Figure 6.9 Two of the 52 vertical clusters

The second parameter, estimated using the "voxel" procedure, is necessary to do horizontal clustering. The first case is displayed well in figure 6.10, and the second case is displayed well in figure 6.11.

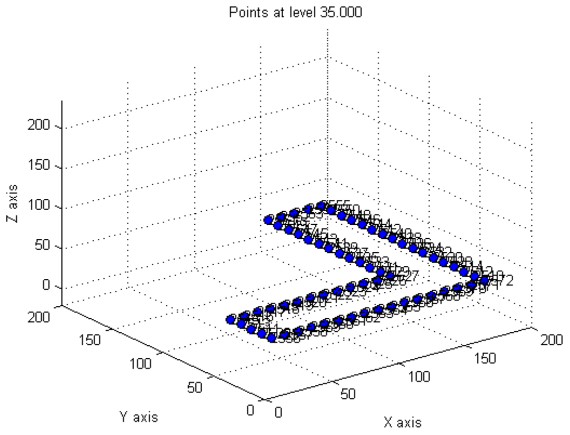


Figure 6.10 One horizontal cluster is needed

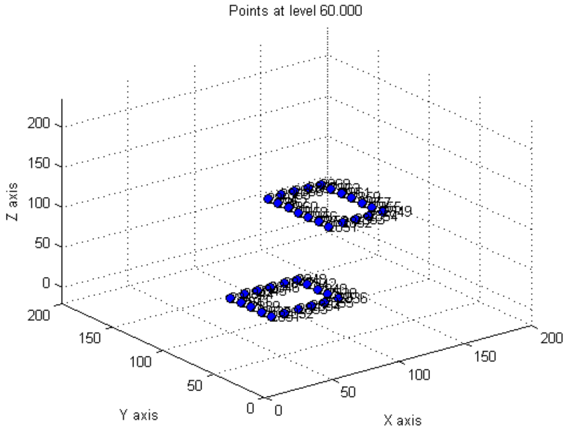


Figure 6.11 Two horizontal clusters are needed

Each horizontal cluster is described by:

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

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

Horizontal ID	Number of points	Height (m)	Point IDs			
1	80	-20.000	1	2	…	441
…	…	…	…			
83	15	235.000	4338	4339	…	4496

Table 6.3 Procedure output with a single horizontal cluster per level

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

Horizontal ID	Number of points	Height	Point IDs			
…	…	…	…			
17	18	55.000	2793	2794	…	2810
18	20	55.000	2811	2815	…	2827
…	…	…	…	…	…	…

Table 6.4 Procedure output with two horizontal clusters (at level 55.000)

There are 83 horizontal clusters and 52 3D plots. If a vertical cluster contains more than one horizontal cluster, the 3D plot contains all the horizontal clusters at that level.

6.4. FEATURES EXTRACTION

The fourth procedure is devoted to distinguishing chains from sowns.

The distinction between sowns and chains is firstly made considering the number of points and secondly, and in the case of ambiguity, the local density.

In the end, if there are some non-classified connections, the type is set as a sown.

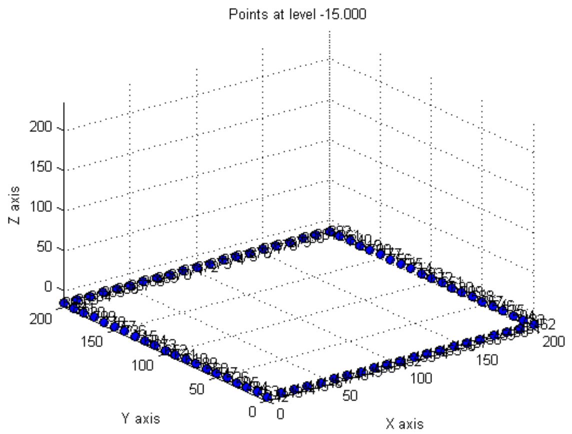


Figure 6.12 One horizontal cluster classified as a chain

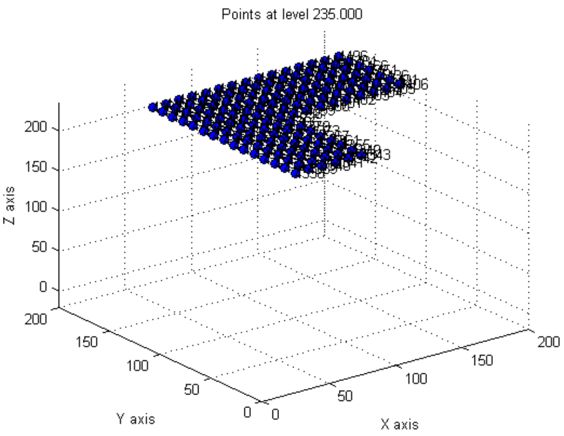


Figure 6.13 One horizontal cluster classified as a sown

Figure 6.14 displays the point cloud with two colors: green if the horizontal cluster represents a sown, blue if it is classified as a chain.

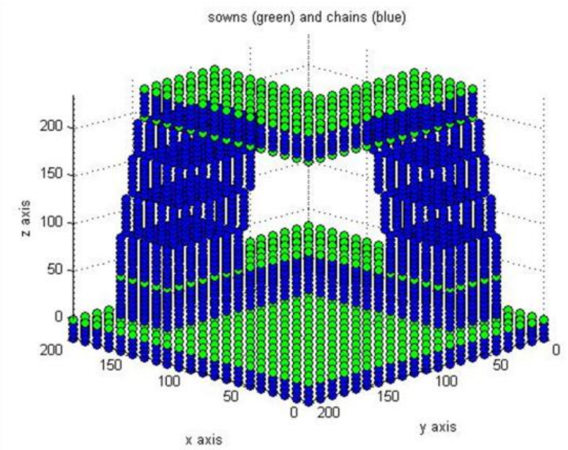


Figure 6.14 Chains (in blue) and sowns (in green)

All horizontal clusters are correctly classified. The information is provided as follows:

- horizontal cluster ID,
- type of clustres, and
- height.

The following table shows the information:

Horizontal ID	Type	Height (m)
1	chain	-20.000
…	…	…
4	chain	-5.000
5	sown	0.000
…	…	…
14	sown	45.000

…	…	…
83	sown	235.000

Table 6.5. Distinction between chain and sown

The levels 0.000, 45.000, 205.000 and 235.000 are classified as sowns, all the others as chains.

6.5. RELATIONAL MATCHING

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 mentioned in the chapter 2, section 5, if the relational matching is not correct, the 3D reconstruct is false and the model created is non-consistent.

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 ID matched to the other horizontal cluster 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, and
- the number of sides matched.

The following table shows the output provided:

Rel_match ID	Hor_ID	Hor_ID	Height (m)	Height (m)	Number of points	Number of points	Point IDs		
1	1	2	-20.000	-15.000	80	80	1	…	440
						80	442	…	881
…	…	…	…	…	…	…	…	…	…
							…	…	…
21	15	17	50.000	55.000	18	18	2755	…	2772
						18	2793	…	2810
22	16	18	50.000	55.000	18	18	2273	…	2789
						18	2881	…	2827
…	…	…	…	…	…	…	…	…	…
							…	…	…
74	82	81	225.000	230.000	56	56	4226	…	4280
						56	4282	…	4336

Table 6.6. Relational matching for chains

The relational matching for sowns is classified by:

- relational matching ID (between the sown and the chain),
- horizontal cluster ID matched to the other horizontal cluster 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, and
- the number of sides matched.

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

Rel_match ID	Hor_ID	Hor_ID	Height (m)	Height (m)	Number of points	Number of points	Point IDs		
1	4	5	-5.000	0.000	80	80	1324	…	1763
						80	1765	…	2123
…	…	…	…	…	…	…	…	…	…
							…	…	…

Table 6.7. Relational matching for sowns

Figure 6.15 shows the point cloud in blue, the relational matching in orange and the perimeters in green.
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.

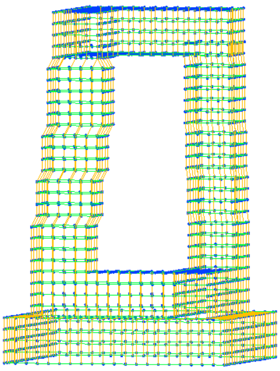


Figure 6.15 Relational matching

6.6. CONTOURS

The sixth procedure is devoted to building closed loops between all chains and all their projections on sowns.
The input files needed are:

- “CCTV_ord.dat”,
- “CCTV_connections.dat”, and
- “CCTV_assembly.dat”.

The output files are:

- “CCTV_closedloop.dat”,
- “CCTV_connections_cla.dat”, and
- “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 and
- 1, for Catmull - Rom loops.

In my 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 m,
- planimetric distance: 15.000 m, and
- planimetric tolerance between points at different levels: 4.000 m.

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 interconnected and each horizontal cluster is finally classified as a chain or as a sown.
At this point it is possible to generate the closed loops, one for each horizontal cluster previously done. All the points must be used only once and all sides, if taken, only once too. All the properties required were explained in chapter 2, section 4.
The output given is divided into two groups: the first one for chains and the second for sown 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, and
- 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, and
- the total number of perimetred points.

The results of the interpolation in Catmull – Rom perimetration 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, and
- y-intercept or x-intercept.

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

Points					
Interpolation ID	**Horizontal cluster ID**	**Point ID**	**X**	**Y**	**Z**
1	1	5001	0.000	0.000	0.000
	…		…	…	…

Table 6.8 New points

Lines								
Interpolation ID	**Horizontal cluster ID**	**Side ID**	**Point ID**			**Type of line**	**Slope**	**x-intersect or y-intersect**
1	1	1	1	2	3	y	0.000	0.000
		2	2	3	4	y	0.000	0.000
	…	…	…	…	…	…	…	…

Table 6.9 Chains interpolation

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

Lines								
Interpolation ID	**Horizontal cluster ID**	**Side ID**	**Point ID**			**Type of line**	**Slope**	**x-intersect or y-intersect**
1	5	1	1765	1766	1767	y	0.000	0.000
		2	1766	1767	1768	y	0.000	0.000
	…	…	…	…	…	…	…	…

Table 6.10 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, and
- total number of intersected points.

If the classic perimetration is set, the connection between the points doesn’t generate other new points. Here two images are attached showing two examples of classic perimeters: the first is for a chain and the second is for a sowns' contour.

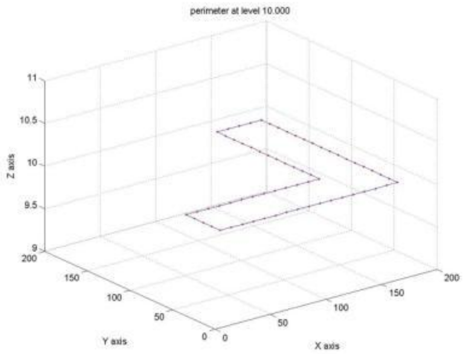


Figure 6.16 Classic perimeter for a chain (10.000 m)

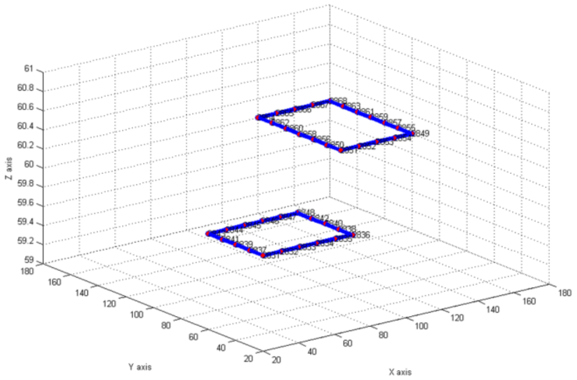


Figure 6.17 Classic perimeter for two chains (60.000 m)

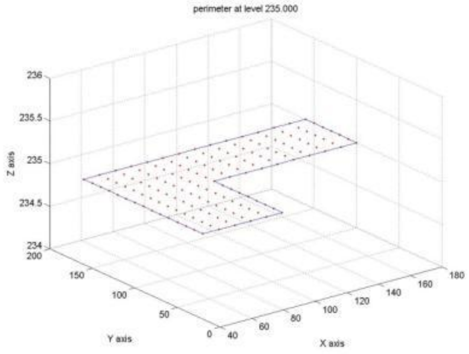


Figure 6.18 Classic perimeter for the contour of a sown (235.000 m)

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

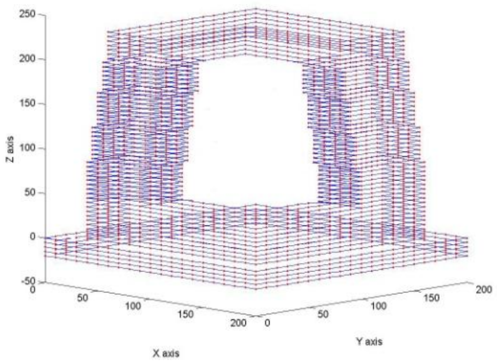


Figure 6.19 Closed paths

The step is completed and it is now possible to move to the next one.

6.7. TRIANGLES

The seventh procedure is devoted to connecting points into triangles. In chapter 2, section 6 this method is examplained.

The points belong to subsequent chains, subsequent sowns, or chains and their projections upon the nearest sowns.

The input files needed in the “triangles” procedure are:

- “CCTV_ord.dat”,
- “CCTV_connections_cla.dat”,
- “CCTV_assembly.dat”,
- “CCTV_closedloops.dat”, and
- “CCTV_contours.dat”.

The output files given are:

- CCTV_triangles.dat,
- CCTV_newpoints.dat.

The three values needed are estimated using “voxel” procedures:

- height interval: 8.000 m,
- planimetric distance: 15.000 m, and
- planimetric tolerance between points at different levels: 4.000 m.

The triangulation must connect all the points belonging to a single horizontal cluster. If one level comprises a single horizontal cluster, the output provided is displayed in figure 6.20.

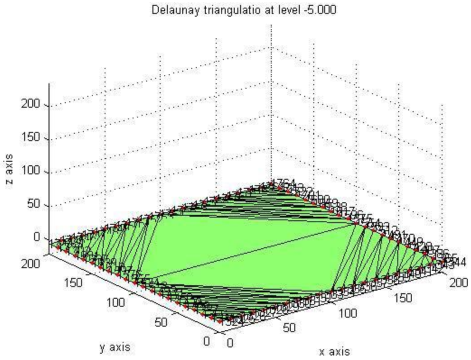


Figure 6.20 Delaunay triangulation (-5.000 m)

If one level comprises more than one horizontal cluster, the output provided is displayed in the following picture.

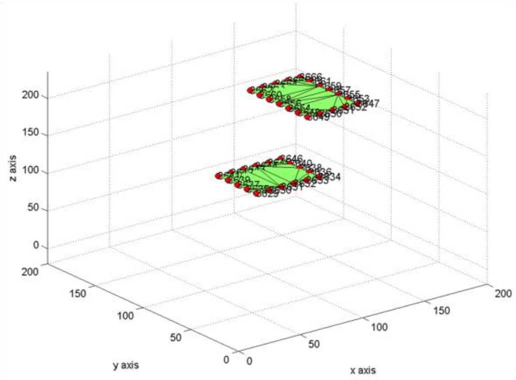


Figure 6.21 Delaunay triangulation (100.000 m)

If one horizontal cluster has a hole, the triangulation must have the same hole. So if a side is found within the hole, it will be deleted. Figure 6.22 shows the initial triangulation (left) and the final one (right). This second one is correct because not only does the horizontal cluster have a hole, but also has the triangulation.

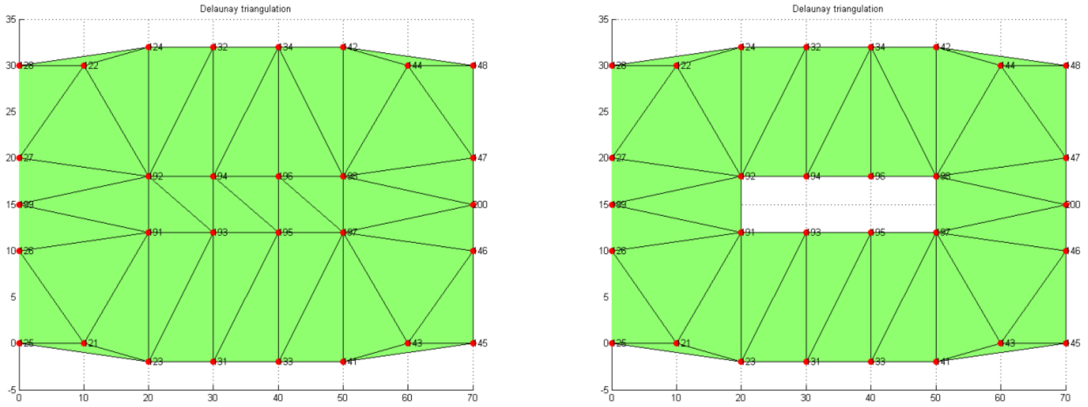


Figure 6.22 Wrong (left side) and right (right side) Delaunay triangulation of a multi-connected shape

Each triangulation is associated with a Voronoi tessellation. For brevity, only the Voronoi tesselation at level -5.000 m:

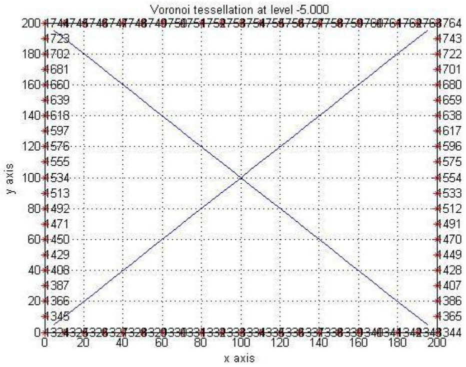


Figure 6. Voronoi tesselation (-5.000 m)

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. Figure 6.24 provides the 3D plot.

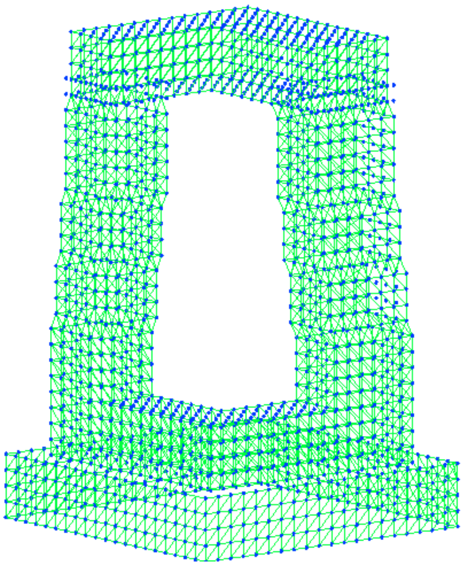


Figure 6.24 Delaunay triangulation

CHAPTER 7

The second case study: an airport in Italy

The input data are composed of two sets of information: the coordinates of the input point cloud and the Aerodrome Reference Code. The input point cloud is defined by the three coordinates of the points and a type identifier. If a point is classified as a light along the axis of the runway, its type is set to 1, otherwise 2. The Aerodrome Reference Code is of code comprised of numbers and letters selected for aerodrome planning purposes in accordance with the characteristics of the airplane for which an aerodrome facility is intended [McGrow-Hill dictionary of aviation]. The Aerodrome Reference Code of this case study is equal to 4E.
The following table shows an extraction of the input data set.

Type	North	East	Height
1	5057764.561	1553696.218	237.558
1	5057768.338	1553681.282	237.627
1	5057771.932	1553666.797	237.610
…	…	…	…
2	5057322.073	1555709.506	233.673
2	5057319.261	1555720.454	233.578
2	5057316.889	1555729.65	233.491
…	…	…	…

Table 7.1 An extract of the input data

The shape of this infrastructure is:

- stellar: starting from one point, to reach all the other points, it is not necessary to go outside the infrastructure,
- concave, and

- single–connected: this means that there are not holes.

The reference system adopted is the Gauss - Boaga. This reference system is used in Italy and figure 7.1 shows its characteristics.

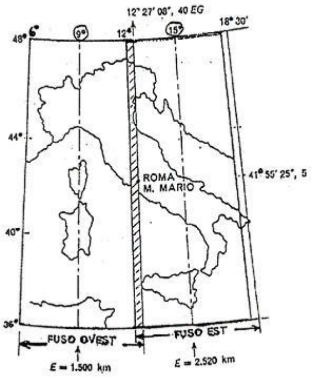


Figure 7.1 Gauss - Boaga reference system

The results of the optimized 3D model will be given in the same reference system in which the coordinates are provided.

An airport is composed of several parts. They are the Clear Way, RESA, center line, runway, runway shoulders, Cleared and Grated Area (CGA) and runway strip.

The elements taken into account to obtain the optimized 3D model are center line, runway, runway shoulders, Cleared and Grated Area (CGA) and runway strip.

Figure 7.2 shows the different parts that comprise an airport.

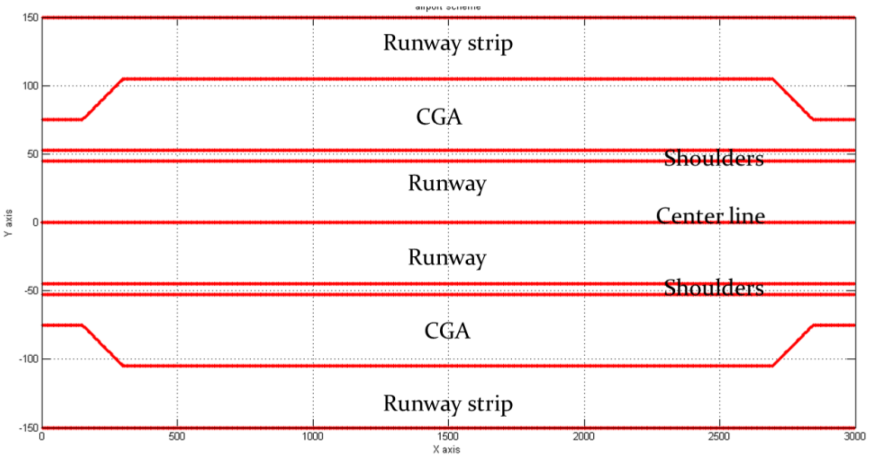


Figure 7.2 airport representation

The optimized 3D model is generated using integrated techniques and the following procedures:

1. check the data set and to provide the first 3D plots,
2. generate the 3D model,
3. analyze the 3D model and the point cloud in terms of clusters,
4. relational matching, or connecting the points that belong to different clusters,
5. connect the points with a Delaunay triangulation,

6. Brückner Profiling, or evaluating the volume of soil to be handled.

The following subchapters explain in detail the different steps.

6.1. OUTLIER DETECTION

The first procedure is devoted to checking the input point cloud for outliers and finding the value of the needed parameter to obtain a 3D model.

As mentioned in the introduction, the input data set has the North, East, Height, and type attributes for each point. If a point is classified as a light along the axis of the runway, its type is equal to one; otherwise its type is equal to two. This is used to identify and to isolate the runway center line.

The runway center line identifies the middle longitudinal section of an airport runway; it has the same distance from the threshold markings located on the runway shoulders. The input file is “Input.txt”, the output file name is “InputData.txt”, and the 3D output plot given is “Input.tif”.

The output file is composed of a list of IDs, North, East, height, and type. The following table shows an extract of the output file.

ID	North	East	Height (m)	Type
1	5057764.561	1553696.218	237.558	1
2	5057768.338	1553681.282	237.627	1
3	5057771.932	1553666.797	237.610	1
…	…	…	…	…
256	5057322.073	1555709.506	233.673	2
257	5057319.261	1555720.454	233.578	2
258	5057316.889	1555729.65	233.491	2
…	…	…	…	…

Table 7.2 An extract of the input file “Input.txt”

The “Input.tif”3D plot is shown in figure 7.3 and it represents the input point cloud. The points classified as lights are in red, the others are in blue.

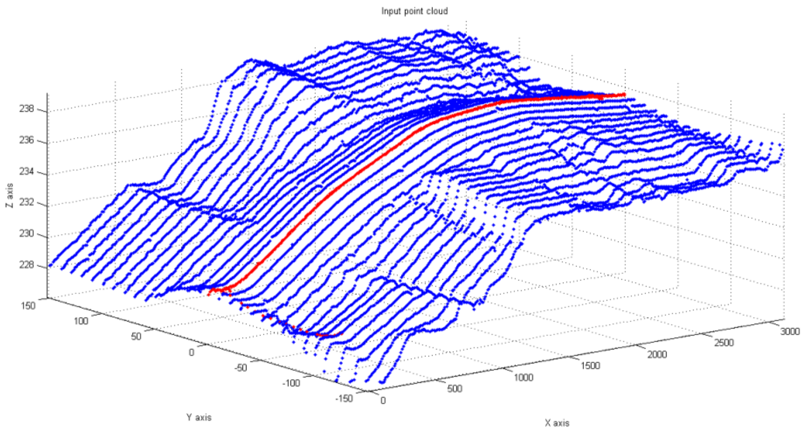


Figure 7.3 Input point cloud

The input file name is “InputData.txt”. The output file names are “InputLight.txt” and “InputOther.txt”. The 3D output plots given are “InputLight.tif” and “InputOther.txt”.

The aim at this point is to identify and to isolate the center line. In order to do this, it is necessary to detect and remove the outliers from the “InputOther.txt” and “InputLight.txt”.

For the former, the input file is “InputOther.txt”, the output file is “InputOtherRev.txt”, and the 3D output plot given is “InputOtherRev.tif”. Because there are no outliers in this set, the file named “InputOther.txt” is equal to “InputOtherRev.txt”.

For the latter, the input file is “InputLight.txt”, the output file is “InputLightRev.txt”, and the 3D output plot given is “InputLightRev.tif”. Figure 7.4 shows the image “IputSpot.tif”. The outliers are highlighted in the blue circles.

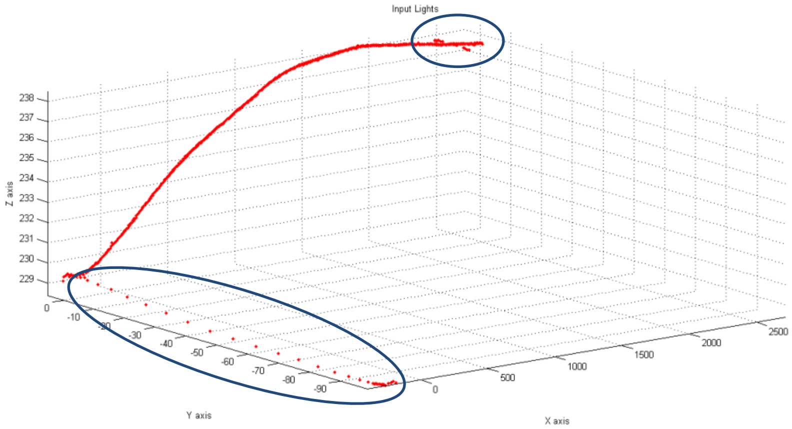


Figure 7.4 Input lights along the axis of the strip

The points highlighted are outliers and they are removed. The others are classified as belonging to the centre line. The following picture shows those points that are used to obtain the optimized 3D model.

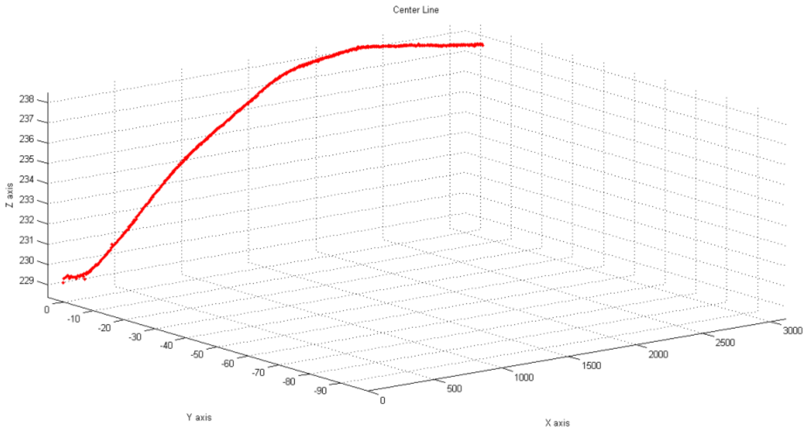


Figure 7.5 Output lights along the axis of the strip

After the outlier detection, it is now possible to identify and isolate the center line.

The input file needed is “InputLightRev.txt”. The output file given is “InputCenterLine.txt”. The 3D output plots given are “InputCenterLine.tif”, “InputOther.tif”, and “InputSets.tif”. Figure 7.6

shows the image “InputSets.tif” with two different colors: the center line (first set) is in red and the other points (second set) are in blue.

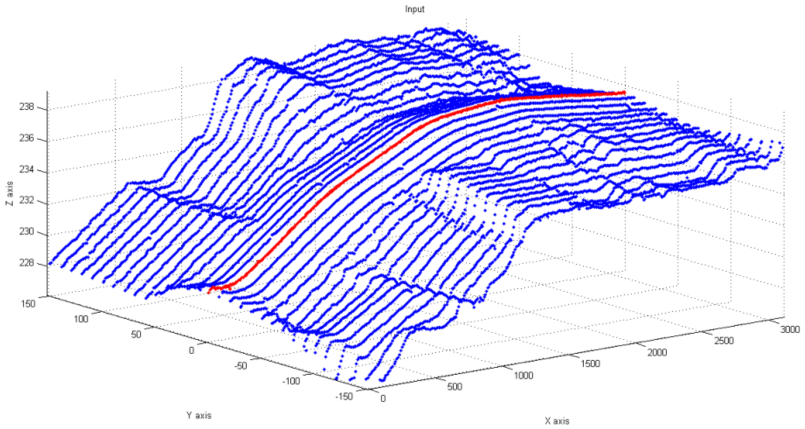


Figure 7.6 Center line in red and the other points in blue

It is also possible to estimate the value of space parameter using the total number of points and the fact that the points are equally spaced along the area. In this case the space parameter is equal to 10 m. Some adaptations must be done because some required widths are lower than 10 meters. Figure 7.7 shows the point cloud and the point of view is set so the space parameter is checked.

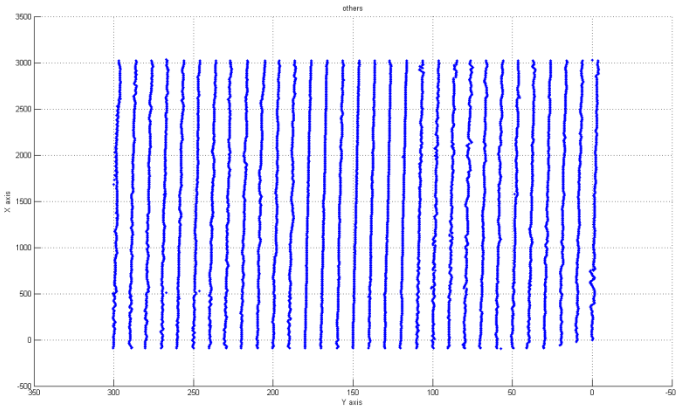


Figure 7.7 Point cloud and its space parameter

The optimize 3D model will be generated starting from the points classified as belonging to the center line and using the space parameter. The following picture provides “InputCenterLine.tif”.

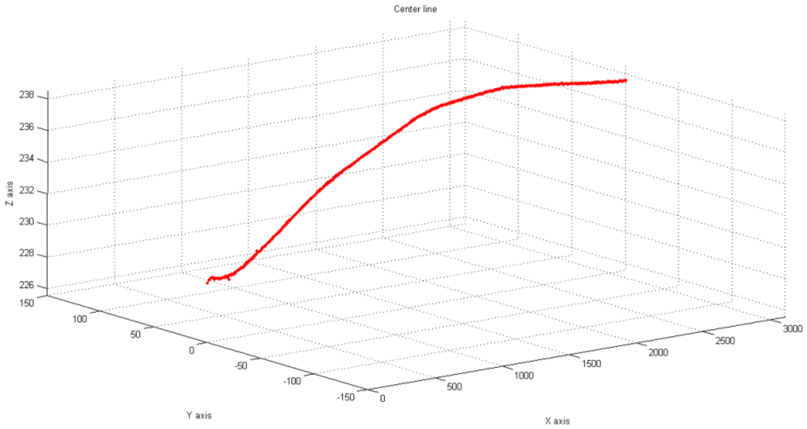


Figure 7.8 Center line

6.2. SYNTHETIC MODEL

6.2.1. Planimetric model

The second procedure is devoted to providing the planimetric model and the 3D model. The former is obtained starting with the “InputCenterLine.txt” file and the 3D model is obtained from the planimetric model.

As mention in the introduction of this chapter, an airport is composed of several parts. The dimensions of each element depend on the Aerodrome Reference Code. In this particular case, the Aerodrome Reference Code is 4E.

For this reason, the prescribed widths are:

- runway: 45 meters,
- runway shoulders: 52.5 meters (45m runway and 7.5m per side),
- Cleared and Grated Area (CGA): variable:
    - Minimum: 75 meters,
    - Maximum: 105 meters, and
- runway strip: 150 meters.

The aim at this point is to obtain a planimetric model with the required widths.

The input file needed is “InputCenterLine.txt”. The output file given is “PlanimetricModel.txt”. The 3D output plots given are “PlanimetricModel.tif” and “DenserPlanimetricModel.tif”.

The parts that comprise the airport are now generated starting from the coordinates of the center line.

The planimetric model is composed of the parts prescribed by the Aerodrome Reference Code. Figure 7.9 shows the “PlanimetricModel.tif” and represents the airport scheme and the different colors are related to the difference parts:

- centre line: in red,
- runway: in blue,
- runway shoulders: in green,

- Cleared and Grated Area (CGA): in black, and
- runway strip: in purple.

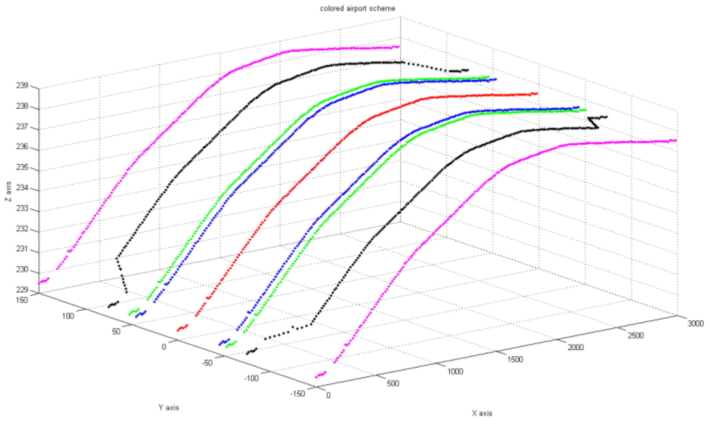


Figure 7.9 Planimetric model

Using the space parameter it is possible to obtain a model characterized by the same density of the input point cloud.

Figure 7.10 (“DenserPlanimetricModel.tif”) represents the 3D model, with the prescribed parts, and the same density of the input point cloud. The meaning of the colors is the same of figure 7.9. The points in yellow are the ones in between the limits.

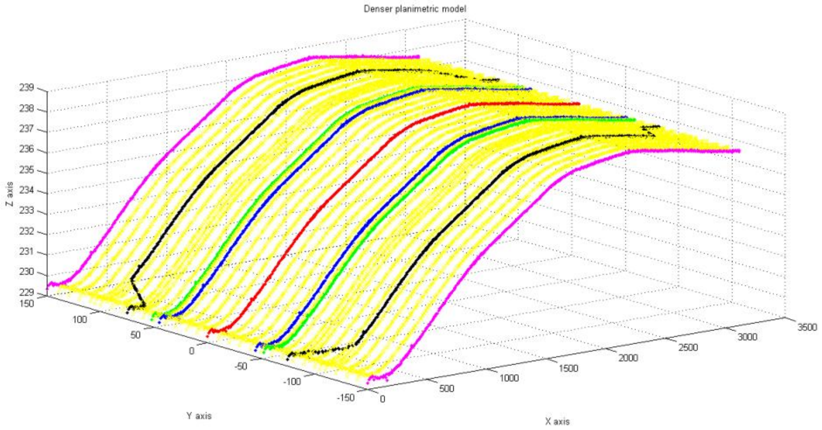


Figure 7.10 Denser planimetric model

The different parts are assigned a different type and the following list provides the classification used:

- centre line: the type is set to 1,
- runway: the type is set to 2,
- runway shoulders: the type is set to 3,
- Cleared and Grated Area (CGA): the type is set to 4, 5, 6, or 7,
- runway strip: the type is set to 8, and
- else: the type is set to 10.

The following table shows an extraction of the file named “3DModel.txt”.

ID	North	East	Height	Type
1	0	0	229.48	1
2	0	45	229.48	2
…	…	…	…	…
5	0	150	229.48	8
6	15	0	229.54	1
…	…	…	…	

Table 7.3 Extract of the “FlatModel.txt” file

6.2.2. 3D model

This step is devoted to providing the 3D model. It is obtained starting from the denser planimetric model, which is shown in figure 7.7.

Each part of the airport is not only characterized by the width, but also by its proper cross slope. The cross slope is expressed in percentage:

$$\frac{\Delta Z}{\Delta \mathrm{Y}} * 100 = slope$$

A cross slope example is displayed in figure 7.11.

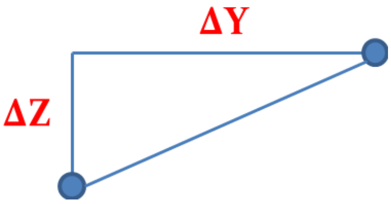


Figure 7.11 Cross slope

The different parts of the airport have different cross slopes and they are related to the Aerodrome Reference Code. This study case is characterized by an A.E.R. equal to 4E, therefore the cross slopes are:

- runway: 1.5% (1:66),
- runway shoulders: 2.5% (1:40),
- Cleared and Grated Area (CGA): 2.5% (1:40), and
- runway strip: 2.5% (1:40).

The input file needed is “DenserPlanimetricModel.txt”, the output file given is “3DModel.txt”, and the 3D output plot given is “3DModel.tif”.

Figure 7.12 shows the 3D airport scheme from two different points of view. This model respects both the widths and the cross slopes prescribed by the Aerodrome Reference Code.

The meaning of the colors and the classification used are the same adopted in “DenserPlanimetricModel.txt”.

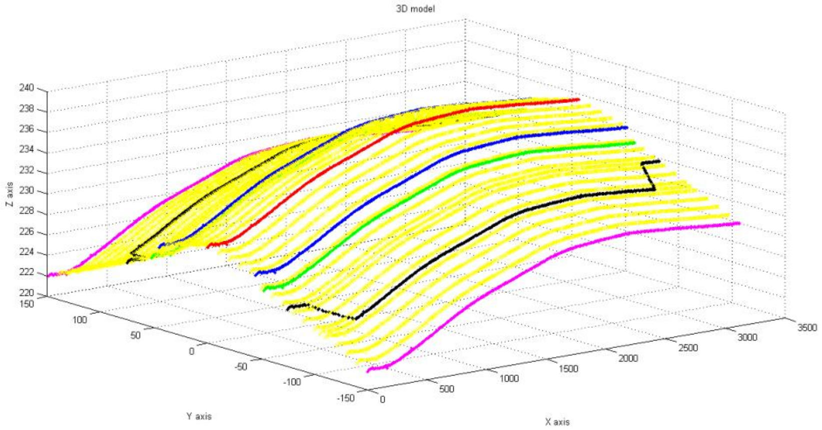


Figure 7.12 3D model

The output file is composed of a list of ID, North, East, Height, and type, as shown in the extract below.

ID	East	North	Height	Type
…	…	…	…	…
23	59.97	52.50	229.27	3
24	59.97	75.00	228.71	4
25	59.97	150.00	226.84	8
26	74.96	0.00	229.64	1
27	74.96	45.00	229.53	2
28	74.96	52.50	229.34	3
…	…	…	…	…

Table 7.4 Extract of the “3DModel.txt” file

At this point it is also possible to generate the different sections of the 3D model. For brevity, figure 7.13 shows the first section. The different colors have the same meaning adopted in “3DModel.tif”.

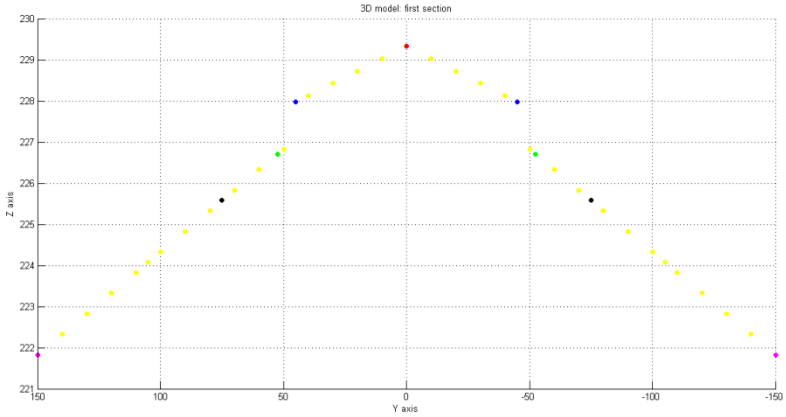


Figure 7.13 First section

At this point it is now possible to move on to the next step.

6.3. CLUSTERS ANALYSIS

The third procedure is devoted to analyzing the data in terms of clusters. As explained in chapter 2, section 3, the word clustering means the operation that assigns a set of observations to subsets called clusters.

The initial assessment is displayed in figure 7.15. It shows two data sets: the 3DModel set and the InputSet. The former is displayed using six colors: the center line in red, the runway in blue, the runway shoulders in green, the clear and grated area in black, the runway strip in purple and the other points in yellow. The latter provides the center line in red and all the other points in blue.

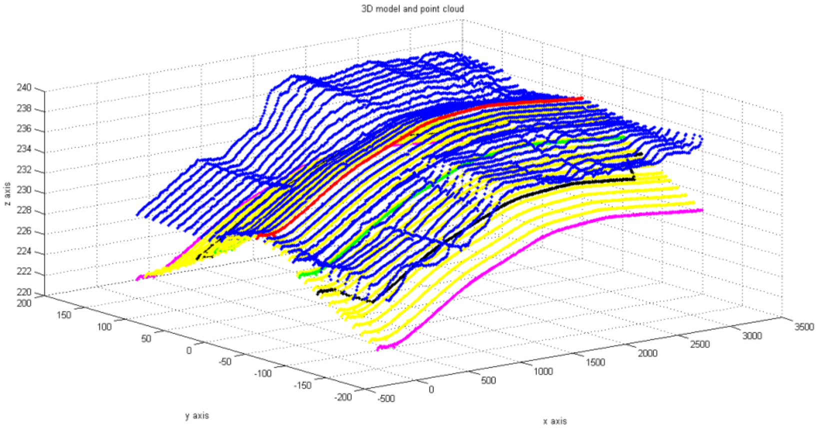


Figure 7.15 3D model and the point cloud

The cluster analysis procedure is done separately for the 3D model and for the input data set.

For the 3D Model, the input file needed is “3DModel.txt”. The output files given are:

- “3DClusters.txt”,
- “CenterLineCluster.txt”,
- “RunwayCluster.txt”,
- “RunwayShouldersClusters.txt”,
- “CGACluster.txt”,
- “RunwayStripCluster.txt”,
- “OtherPointsCluster.txt”.

The 3D output plots given are:

- “3DClusters.tif”,
- “CenterLineCluster.tif”,
- “RunwayCluster.tif”,

- “RunwayShouldersCluster.tif”,
- “CGACluster.tif”,
- “RunwayStripCluster.tif”, and
- “OtherPointsCluster.tif”.

Figure 7.16 shows the image “3DClusters.tif”.

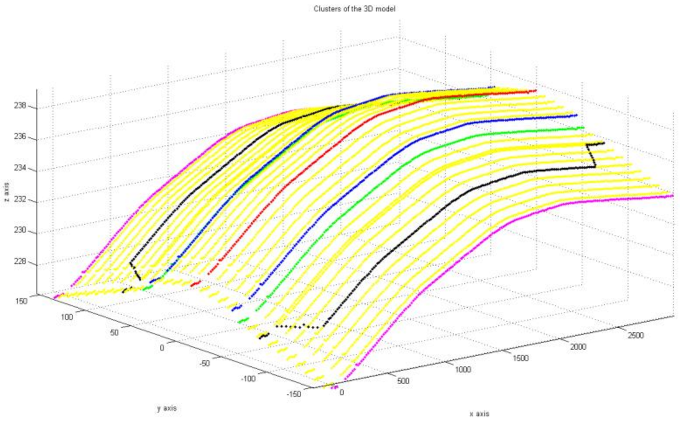


Figure 7.16 Clusters of the 3D model

Figure 7.17 shows the image “CenterLineCluster.tif”. It only contains the points belonging to the center line.

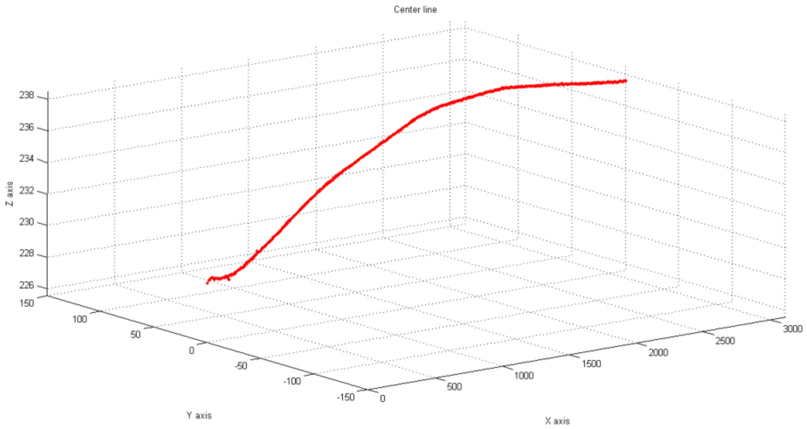


Figure 7.17 Center line cluster

Figure 7.18 displays the image “RunwayCluster.tif”. It only contains the points belonging to the runway.

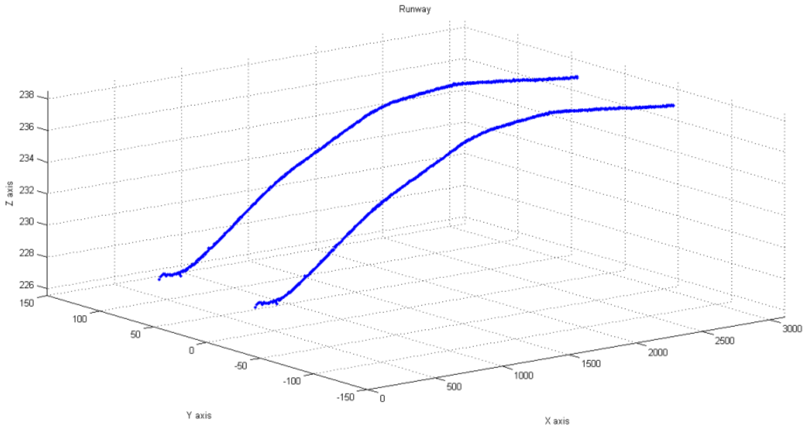


Figure 7.18 Runway cluster

Figure 7.19 shows the image “RunwayShouldersCluster.tif”. It only displays the points belonging to the runway shoulders.

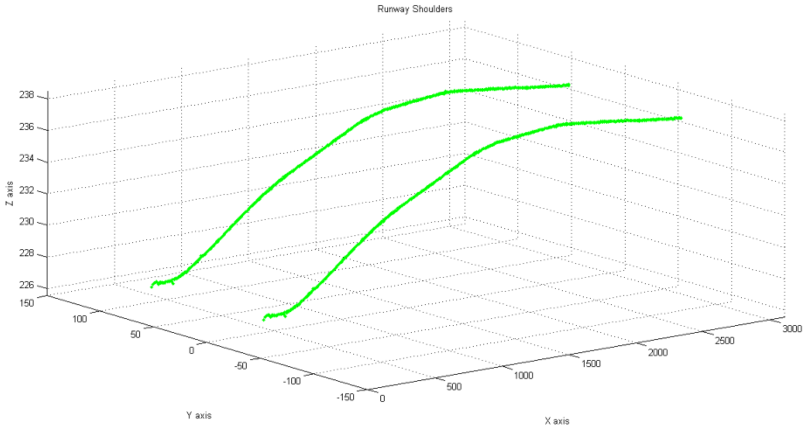


Figure 7.19 Runway shoulder cluster

Figure 7.20 displays the image “CGSCluster.tif”. It only shows the points classified as belonging to the clear and grated area.

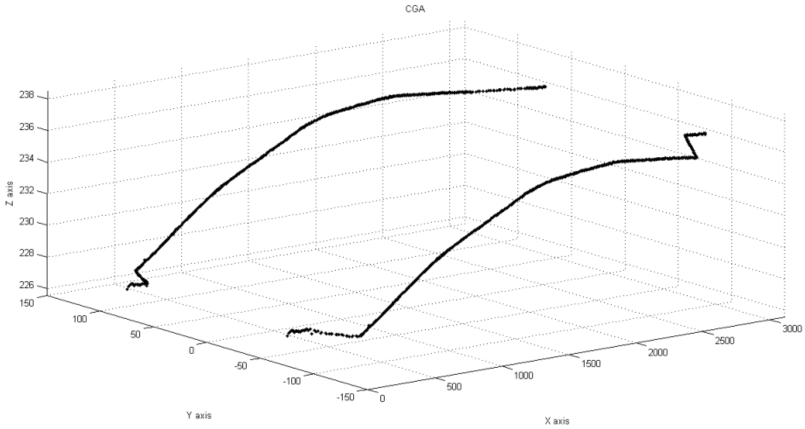


Figure 7.20 CGA cluster

Figure 7.21 shows the image “StripCluster.tif”. It displays the points belonging to the runway strip.

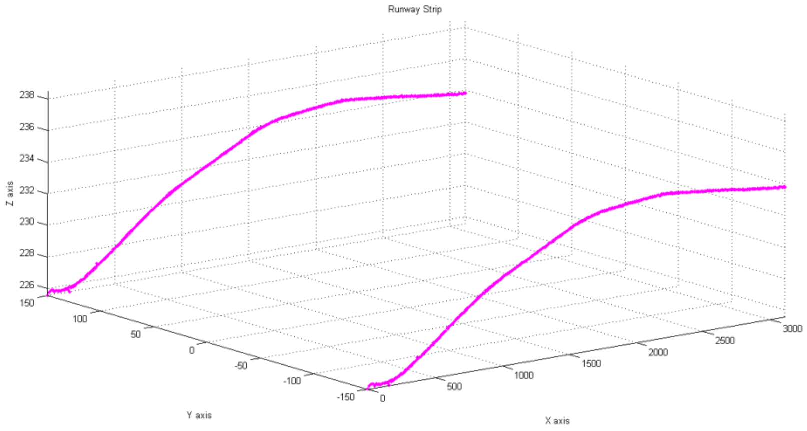


Figure 7.21 Runway strip cluster

Figure 7.22 shows the image “OtherClusters.tif”. It displays all the points in between the limits prescribed by the Aerodrome Reference code.

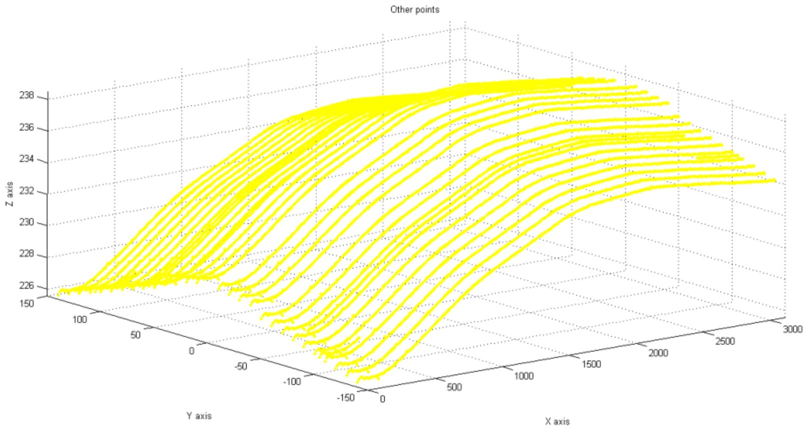


Figure 7.22 Other points

For the cluster analysis of the point cloud, it is necessary to cluster the “InputSets.txt”. Because of the density of the point cloud, it is impossible to detect the following parts of the airport:

- runway: it is at 45m from the center line. There are data at 40 m and at 50 m.
- runway shoulders: it is at 52.5 m from the center line. There are data at 50 m and at 60 m.
- CGA: it is between 75 m and 105 m from the center line. There are data at 70 m, 80 m, 90 m, 100 m, and 110 m.

For this reason, it is necessary to interpolate the file named “InputSet.txt”.

- runway: it is obtained as the mean between the data at 40 m and 50 m.
- runway shoulders: it is obtained as the weight mean between the data at 50 m and 60 m.

- CGA: it is obtained as the mean between the data at 70 m and 80 m, and between the data at 80 m and 90 m.

The input file needed is “InputSet.txt”. The output files given are:

- “Runway.txt”,
- “RunwayShoulders.txt”,
- “NarrowCGA.txt”, and
- “WideCGA.txt”.

The 3D output plots given are:

- “Runway.tif”,
- “RunwayShoulders.tif”,
- “NarrowCGA.tif”, and
- “WideCGA.tif”.

Figure 7.23 shows the points with the y-coordinates equal to 40 m in blue, the y-coordinates equal to 50 m, and the y-coordinates equal to 45 m in red.

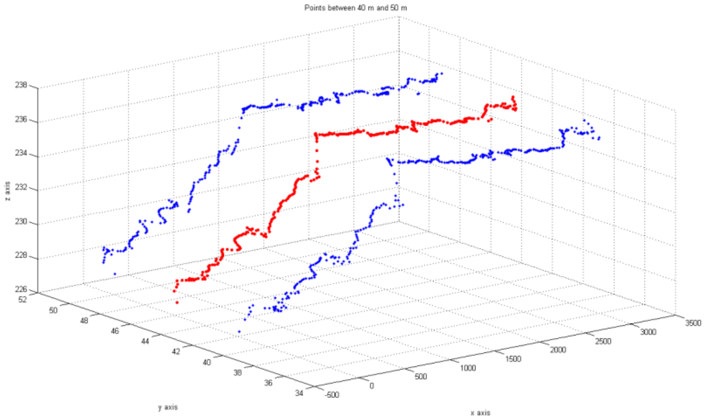


Figure 7.23 Points at 40 m (in blue), 45 m (in red), and 50 m (in blue)

Figure 7.24 shows the points with the y-coordinates equal to 50 m in blue, the y-coordinates equal to 60 m, and the y-coordinates equal to 52.5 m in red.

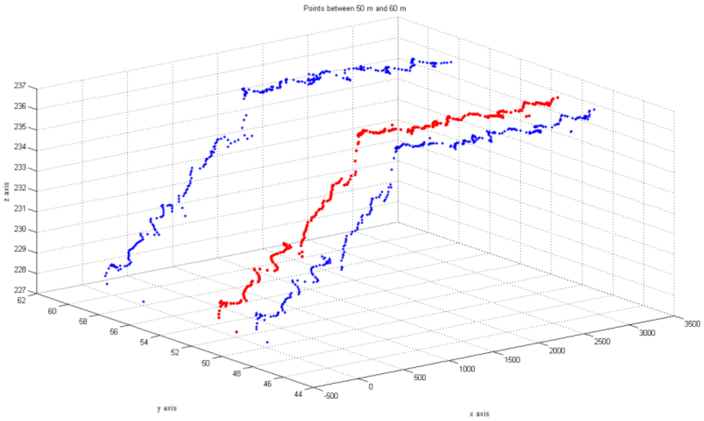


Figure 7.24 Points at 50 m (in blue), 52.5 m (in red), and 60 m (in blue)

Figure 7.25 shows the points with the y-coordinates equal to 70 m in blue, the y-coordinates equal to 80 m, and the y-coordinates equal to 75 m in red.

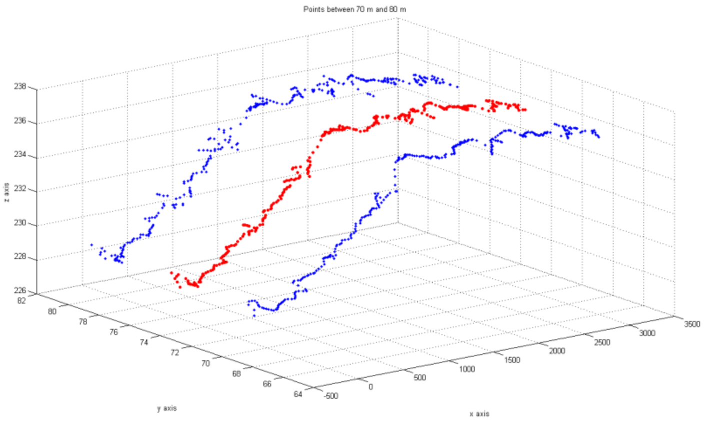


Figure 7.25 Points at 70 m (in blue), 75 m (in red), and 80 m (in blue)

Figure 7.26 shows the points with the y-coordinates equal to 100 m in blue, the y-coordinates equal to 110 m, and the y-coordinates equal to 105 m in red.

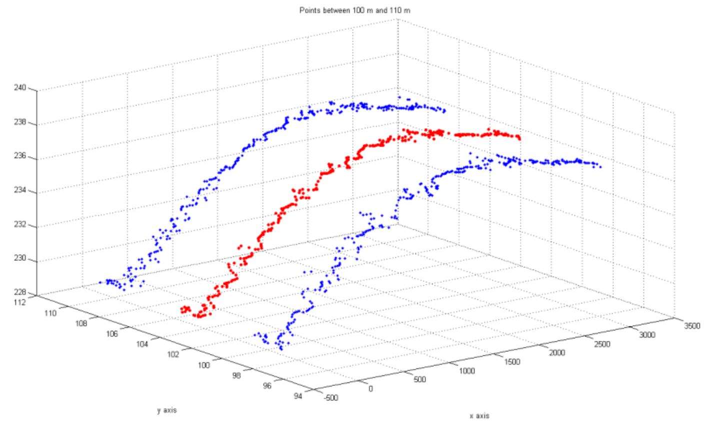


Figure 7.26 Points at 100 m (in blue), 105 m (in red), and 110 m (in blue)

At this point it is possible to do the cluster analysis of the point cloud. The input files needed are:

- “InputOtherRev.txt”,
- “Runway.txt”,
- “RunwayShoulders.txt”,
- “NarrowCGA.txt”, and
- “WideCGA.txt”.

The output files given are:

- “InputClusters.txt”,
- “InputCenterLineCluster.txt”,
- “InputRunwayCluster.txt”,
- “InputRunwayShouldersClusters.txt”,
- “InputCGACluster.txt”,
- “InputRunwayStripCluster.txt”, and
- “InputOtherPointsCluster.txt”.

The 3D output plots given are:

- “InputClusters.tif”,
- “InputCenterLineCluster.tif”,
- “InputRunwayCluster.tif”,
- “InputRunwayShouldersClusters.tif”,
- “InputCGACluster.tif”,
- “InputRunwayStripCluster.tif”, and
- “InputOtherPointsCluster.tif”.

Figure 7.27 shows the final clusters:

- center line cluster: the points are displayed in red,
- runway cluster: the points are displayed in blue,
- runway shoulders cluster: the points are displayed in green,
- CGA: the points are displayed in black,
- runway strip: the points are displayed in purple, and
- all the other points are displayed in yellow.

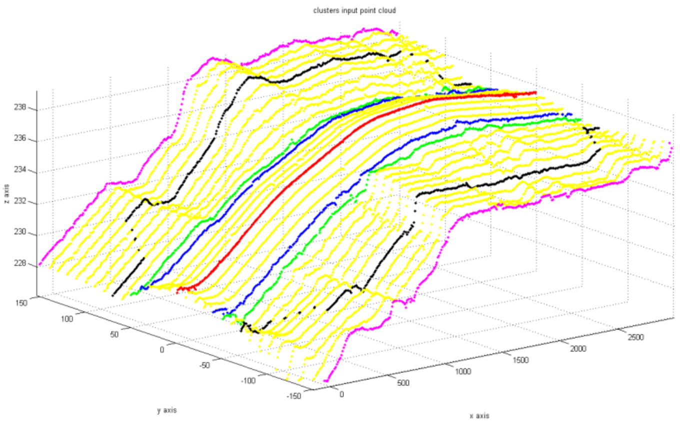


Figure 7.27 Clustered input point cloud

Each point has a different type, depending on the cluster it belongs to. The following list provides the different types:

- type is equal to 1 if the point belongs to the centre line cluster,
- type is equal to 2 if the point belongs to the runway cluster,
- type is equal to 3 if the point belongs to the runway shoulders cluster,
- type is equal to 4, 5, 6, and 7 if the point belongs clear and grated area,
- type is equal to 8 if the point belongs to the runway strip cluster, and
- type is equal to 0 if the point doesn't belong to any of the previous clusters.

The following table shows an extract of the “InputClusters.txt” file.

ID	North	East	Height	Type
…	…	…	…	…
9145	2303.47	-110.09	235.91	7
9146	2312.47	-110.03	235.94	7
9147	2323.33	-110.04	235.94	7
9150	2353.90	-109.87	235.91	7
…	…	…	…	…

Table 7.5 Extract of the “InputClusters.txt” file

It is now possible to compare the points that belong to the same clusters but to two different models. The following six pictures show the points with dots if they belong to the 3D model, with rhombs if they belong to the point cloud file. As usual, the center line cluster is displayed in red, the runway in blue, the runway shoulders in green, the CGA in black, the runway strip in purple, and the others points in yellow.

6.4. RELATIONAL MATCHING

At this point, the data set is analyzed in terms of relational matching. As mentioned in the chapter 2, section 5, if the relational matching is non-correct, it is impossible to optimize the 3D model. In this specific case, a perfect mapping between each point is attempted so the matching type is a one to one.

Several relational matching iterations are completed. The first one involves all the points of the point cloud and it is done in horizontally. For example the matching is done between all the points with y-coordinate equal to 150 m with the points with y-coordinate equal to 140 m. Then with all the points with y-coordinate equal to 140 m with the points with y-coordinate equal to 130 m and so on.

The following sketch will clarify the method. The points are displayed in orange, the point IDs in blue, and the relational matching in dashed red arrows.

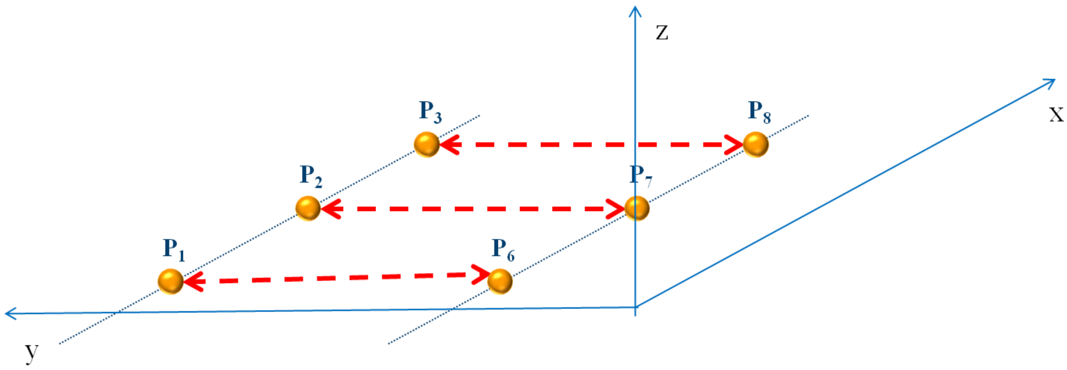


Figure 7.28 Horizontal relational matching

The input file needed is “InputCluster.txt”, the output file give is “InputRelationalMatching.txt”, and the 3D output plots given are as many as the number of the pairs of coordinates. In this case study the software provides 31 3D plots.

The following picture is one of these 31 3D plots. The points with the y-coordinate equal to -150.000 m are displayed in blue, the points with the y-coordinate equal to -140.000 m are displayed in red.

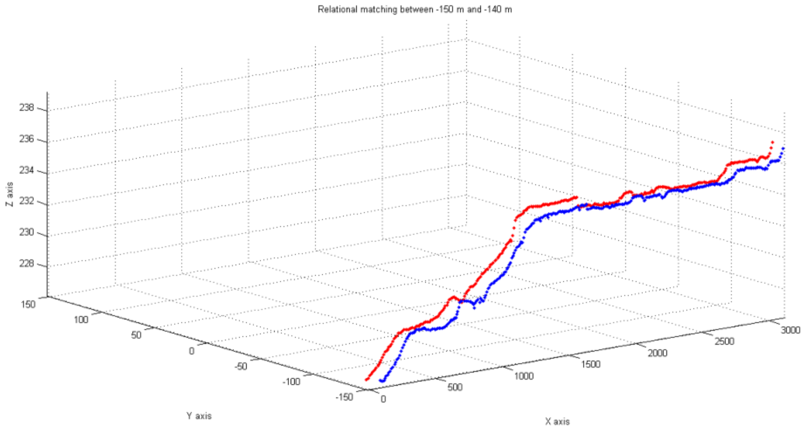


Figure 7.29 Horizontal relational matching: point cloud -150 m and -140 m

The information is listed in this way:

- The ID of the point
- Its North, East, and height coordinate
- The ID of the point that matches the other point
- Its North, East, and height coordinate

The following table shows an example of the output extract:

ID	North	East	Height (m)	ID	North	East	Height (m)
...	...	...	...	...	...	...	...
889	2712.88	98.86	236.35	3004	2711.61	100.60	238.41
904	2757.81	89.88	236.54	3055	2751.71	90.18	238.14
914	2787.79	83.88	236.66	3522	2793.68	81.11	237.34
...	...	...	...	...	...	...	...

Table 7.6 Extract of the output

The second relational matching involves all the points of the 3D model and it is done horizontally. For example the matching is done between all the points with y-coordinate equal to 30 m with the points with y-coordinate equal to 20. Then with all the points with y-coordinate equal to 20 m with the points with y-coordinate equal to 10 m and so on.

The input file needed is “3DCluster.txt”, the output file give is “3DRelationalMatching.txt”, and the 3D output plots given are as many as the number of the pairs of coordinates. In this case study the software provides 31 3D plots.

The following picture is one of these 31 3D plots. The points with the y-coordinate equal to -150.000 m are displayed in blue, the points with the y-coordinate equal to -140.000 m are displayed in red.

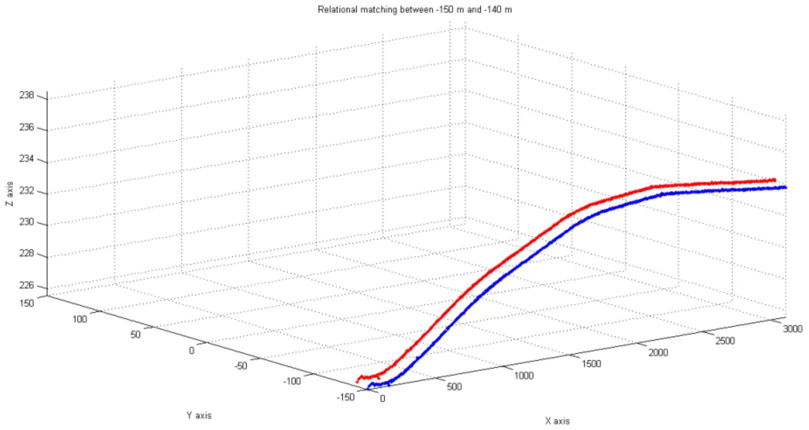


Figure 7.30 Horizontal relational matching

The information is listed in the same way as previously for relational matching.

The third type of relational matching is done vertically. Points with the same y-coordinate that belong to the point cloud are matched with the same y-coordinate that belong to the 3D model.

The following example will clarify the method. The points are displayed in orange, the point IDs in blue, the relational matching in dashed green arrows.

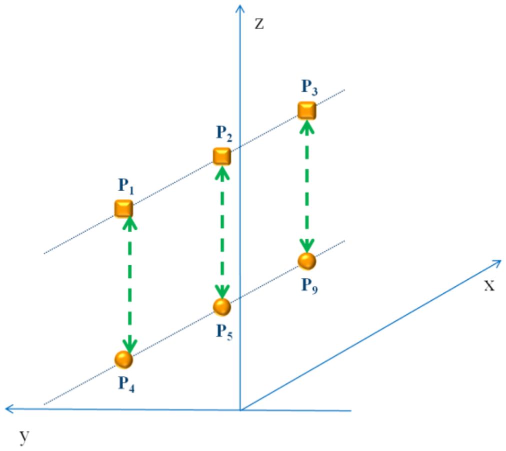


Figure 7.31 Vertical relational matching

The input files needed are “InputCluster.txt” and “3DClusters”, the output file given is “InputModelRM.txt”, and the 3D output plots given are as many as the number of the pairs of coordinates. In this case study the software provides 31 3D plots.
The following picture is one of these 31 3D plots. The points belonging to the model are displayed in blue, while the points belonging to the point cloud are displayed in red.

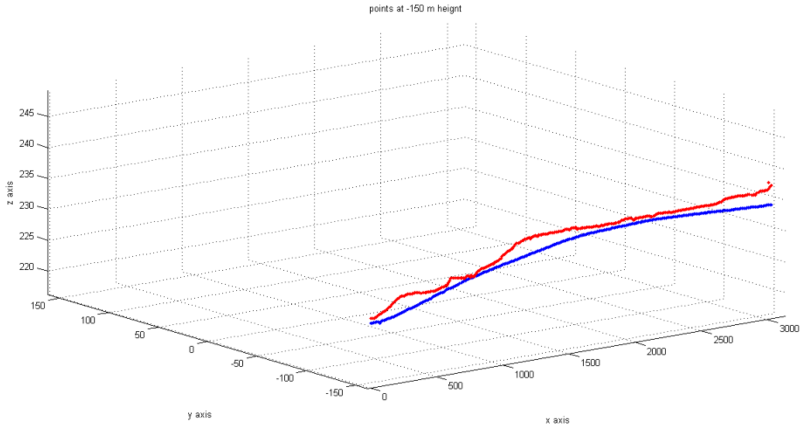


Figure 7.32 Horizontal relational matching: point cloud -150 m and -140 m

The information is listed in this way:

- The ID of the point,
- Its North, East, and height coordinate,
- The set the point belongs (point cloud or 3D model),
- The ID of the point that matches the other point,
- Its North, East, and height coordinate, and
- The set the point belongs (point cloud or 3D model).

The following table shows an example of the output extract:

ID	North	East	Height (m)	Set	ID	North	East	Height (m)	Set
...	...	...	...		...	...	...	...	
50920	91.69	29.86	229.02	3D model	5092	91.69	29.86	229.00	Point cloud
50930	101.06	29.86	229.03	3D model	5093	101.06	29.89	229.05	Point cloud
...	...	...	...		...	...	...	...	

Table 7.7 Extract of the output

The last three types of relational matching can be combined to form a set of four points. Two points belong to the 3D model, two points to the point cloud. The four of them are connected using relational matching. A sketch will clarify the procedure. The points classified as belonging to the point cloud are in light blue, the points classified as belonging to the 3D model are in orange, the point IDs are in black, the horizontal relational matching is in red dashed arrows, and the vertical relational matching is in red dashed arrows.

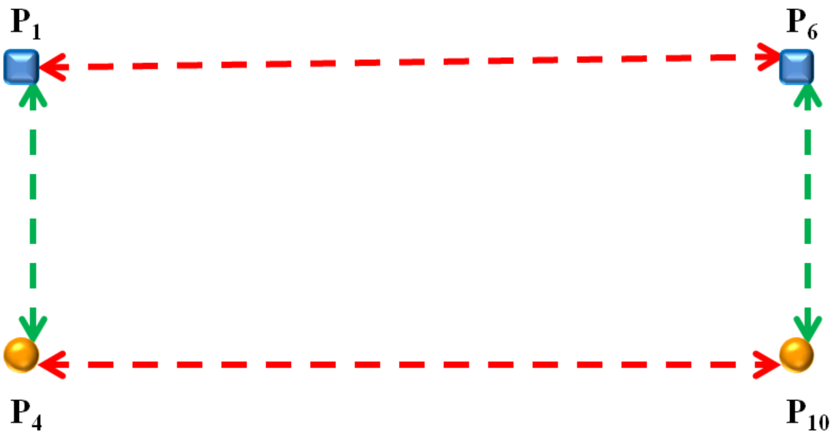


Figure 7.33 The set of four points connected with the different relational matching

These types of relational matching will be used to obtain the area diagrams.
The fourth type of relational matching is done vertically, considering the cluster analysis previously done. Points that belong to the point cloud and are classified as belonging to the same cluster are matched with the points that belong to the 3D model and are classified as belonging to the same cluster.
The input files needed are:

- “CenterLineCluster.txt”,
- “RunwayCluster.txt”,
- “RunwayShouldersClusters.txt”,
- “CGACluster.txt”,
- “RunwayStripCluster.txt”,
- “InputCenterLineCluster.txt”,

- “InputRunwayCluster.txt”,
- “InputRunwayShouldersClusters.txt”,
- “InputCGACluster.txt”, and
- “InputRunwayStripCluster.txt”.

The output files given are:

- “CenterLineRM.txt”,
- “RunwayRM.txt”,
- “RunwayShouldersRM.txt”,
- “CGARM.txt”, and
- “RunwayStripRM.txt”.

The information is as listed as it is done for the previous type of relational matching.
The following picture shows the relational matching between the points classified as belonging to the Runway centre line cluster. The points are displayed in red dots if they belong to the 3D model, in red diamonds if they belong to the point cloud.

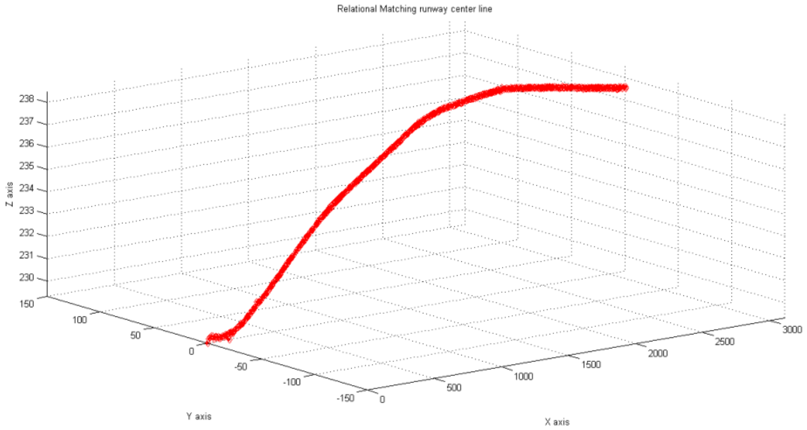


Figure 7.34 Relational matching – Runway center line

The following picture shows the relational matching between the points classified as belonging to the Runway cluster. The points are displayed in blue dots if they belong to the 3D model, in blue diamonds if they belong to the point cloud.

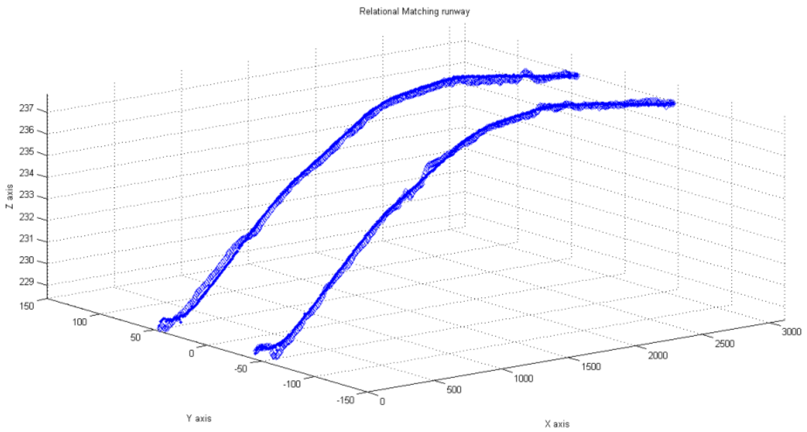


Figure 7.35 Relational matching – Runway

The following picture shows the relational matching between the points classified as belonging to the Runway Shoulders cluster. The points are displayed in green dots if they belong to the 3D model, in green diamonds if they belong to the point cloud.

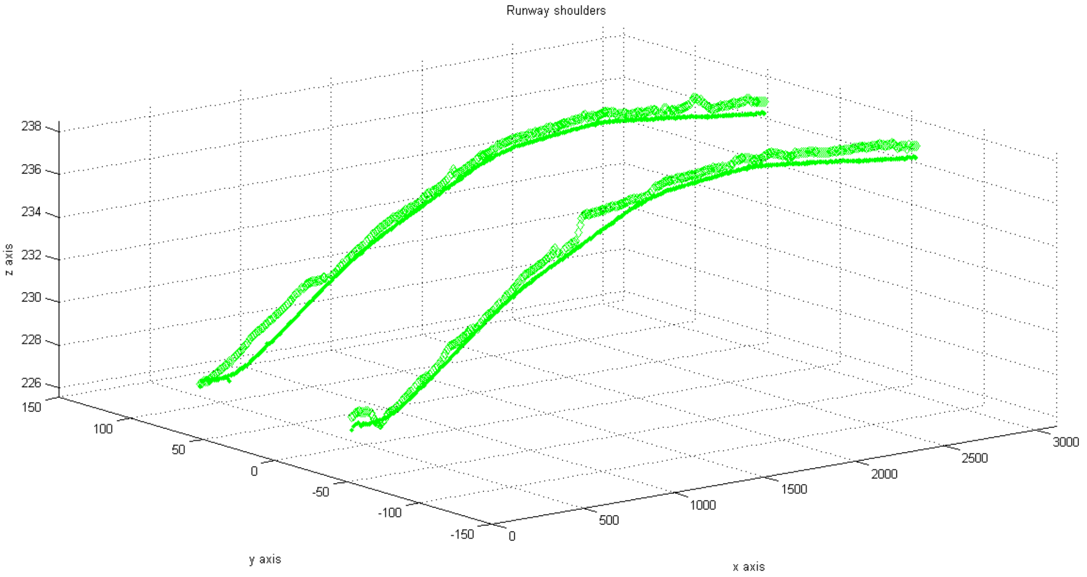


Figure 7.36 Relational matching – Runway shoulders

The following picture shows the relational matching between the points classified as belonging to the Clear and Grated Area. The points are displayed in black dots if they belong to the 3D model, with black diamonds if they belong to the point cloud.

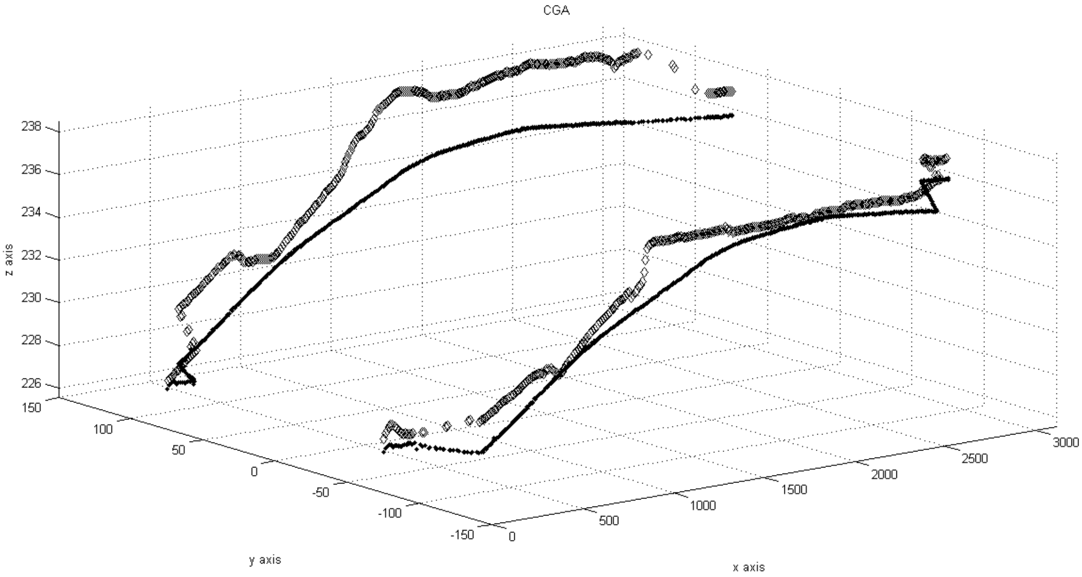


Figure 7.37 Relational matching – Clear and Grated Area

The following picture shows the relational matching between the points classified as belonging to the Runway. The points are displayed in purple dots if they belong to the 3D model, with purple diamonds if they belong to the point cloud.

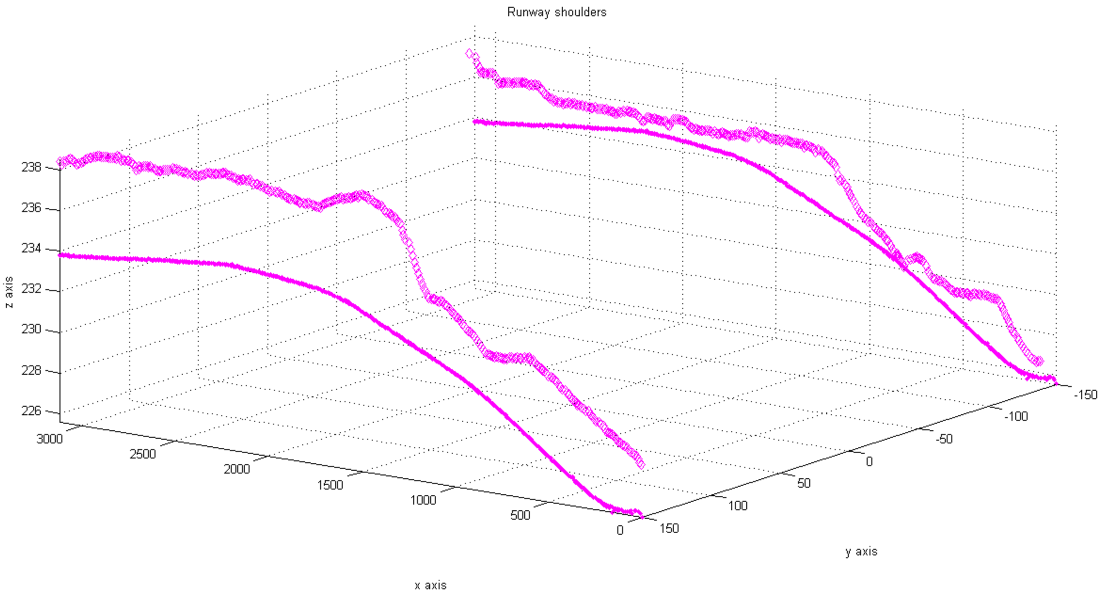


Figure 7.38 Relational matching – Runway strip

6.5. DELAUNAY TRIANGULATION

After the relational matching procedure, the fifth procedure is devoted to connecting the data with Delaunay triangulation. Its properties are described in chapter 2, section 6.

It is necessary to apply Delaunay triangulation in two different ways. The first way is applied horizontally to the point cloud and to the 3D model. The second method is applied to sets of four points. Those sets are generated with the first three types of relational matching earlier described.

For the first type of Delaunay triangulation, the input files needed are “InputClusters.txt” and “3DModel.txt”; the output files given are “InputTriangulation.txt” and “ModelTriangulation.txt”; the 3D output plots given are “InputTriangulation.tif” and “ModelTriangulation.tif”.

Figure 7.39 shows 3D model triangulated. The center line cluster is displayed in red, the runway in blue, the runway shoulders in green, the CGA in black, the runway strip in purple, the other points in yellow, and Delaunay triangulation assumes different colors, depending on the height of the points.

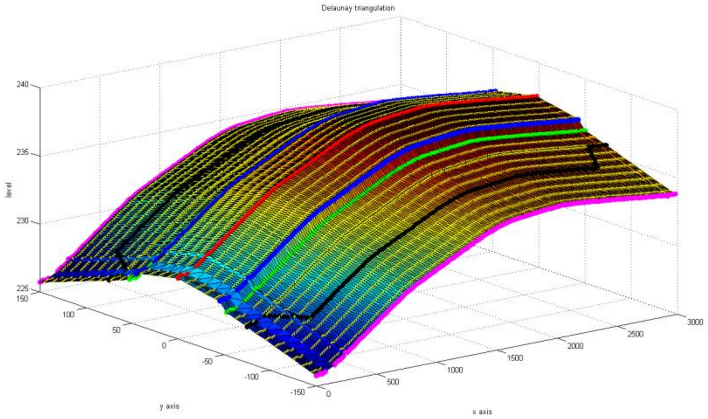


Figure 7.39 Delaunay triangulation of the 3D model

Figure 7.40 shows the point cloud triangulated. The center line cluster is displayed in red, the runway in blue, the runway shoulders in green, the Clear and Grated Area in black, the runway strip in purple, the other points in yellow, and Delaunay triangulation assumes different colors, depending on the height of the points.

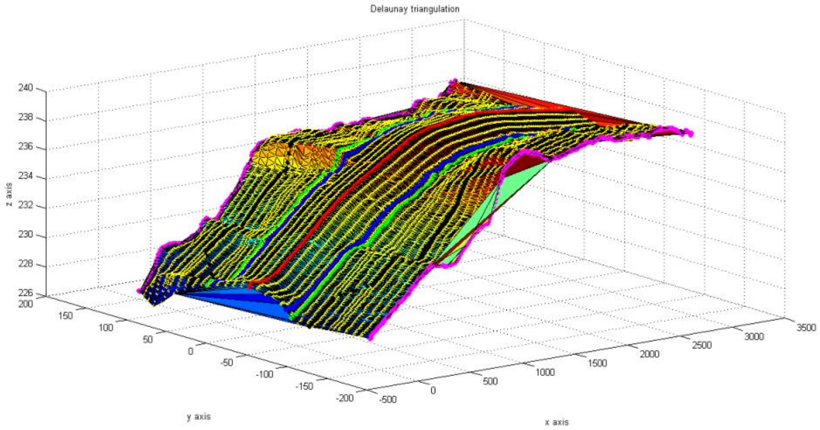


Figure 7.40 Delaunay triangulation of the point cloud

For the second type of Delaunay triangulation, the input file needed is “InputModelRM.txt”, the output file given is “DelaunayTriangulationVertivcal.txt” and 3D output plots, one for each group of four points.

In the following picture, one of those 3D plots is displayed. The four points are in blue and the triangulation is in red.

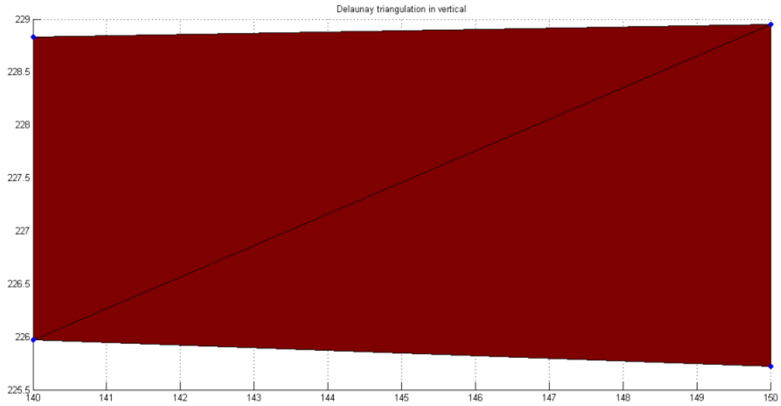


Figure 7.41 Delaunay triangulation in vertical

This type of vertical Delaunay triangulation will be used in the following step.

6.6. STABILIZED ERONE AND THE AREA DIAGRAMS

After the triangulation method, the sixth procedure is devoted to obtaining the area diagrams. To estimate the value of each section, the sum of the areas of two triangles is required. The values are obtained using Erone formula. Its properties are described in chapter 2, section 9.
The applied formula is the following:

$$A = \frac{1}{4} * \sqrt{\big(a + (b + c)\big) * \big(c - (a + b)\big) * \big(c + (a - b)\big) * \big(a + (b - c)\big)}$$

It requires the preparation of the sides so $a \geq b \geq c$ and the brackets are necessary to obtain the required stability.
The input file needed is “DelaunayTriangulationVertivcal.txt” and the output file given is “DiagramAreas.txt”. The information is listed in this way:

- The area of the section,
- The x-coordinate,
- The initial y-coordinate, and
- The final y-coordinate.

The following table is an extract of the output:

Area (m^2)	X-coordinate	Initial y-coordinate	Final y-coordinate
…	…	…	…
47.05	2880.000	140.000	150.000
19.30	2890.000	140.000	150.000

19.73	2900.000	140.000	150.000
…	…	…	…

Table 7.8 Extract of the diagram area

All the areas are positives. It implies there are no fills; the airport will be in cut sections.

The following pictures provide different examples of area diagrams. The x-axis coincides with the axis of the runway, the y-axis displays the area.

The first example is an area diagram of the sections between 140 m and 130 m. The minimum area is 15,43 m^2, starting from the center line to 850 m; the maximum area is 59,38 m^2, starting from the center line to 1490 m. All the areas are positive.

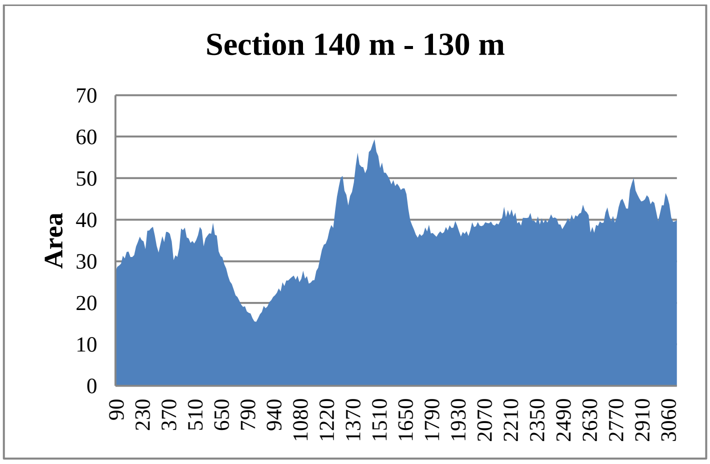


Figure 7.42 Diagram of the area section #2

The second example is an area diagram of the section between 80 m and 70 m. The minimum value of the area is 2,03 m^2 at 810 m from the beginning of the center line, the maximum value is 20,89 m^2 at 2760 m from the beginning of the center line. All the areas are positive.

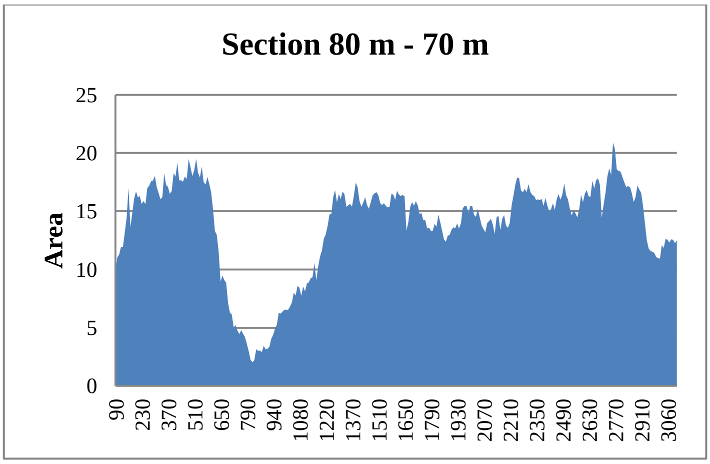


Figure 7.43 Diagram of the area, section #8

The third example is an area diagram of the section between 60 m and 50 m. The minimum value of the area is 1,42 m^2 at 720 m from the beginning of the center line, the maximum value is 12,27 m^2 at 420 m from the beginning of the center line. All the areas are positive.

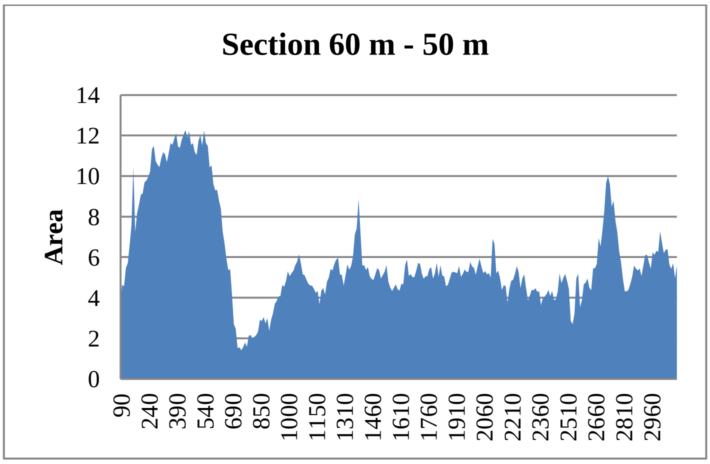


Figure 7.44 Diagram of the area, section #10

The third example is the area diagrams of the section between 20 m and 10 m. The minimum value of the area is 0,05 m^2 at 450 m from the beginning of the center line, the maximum value is 1,83 m^2 at 160 m from the beginning of the center line. All the areas are positive.

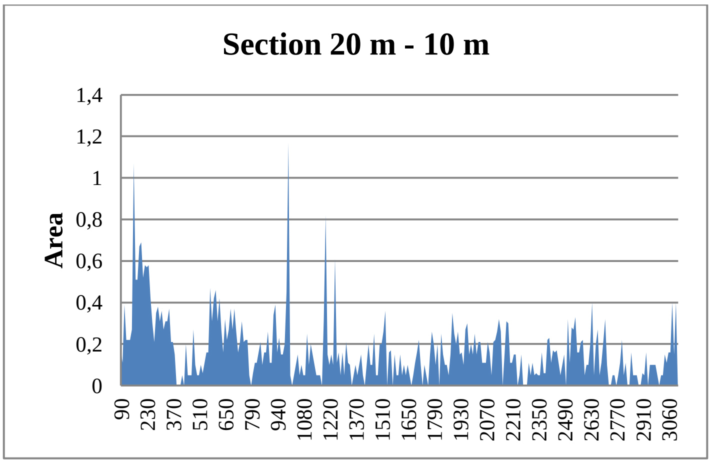


Figure 7.45 Diagram of the area, section #15

All the other area diagrams are positive too. Now that this step is completed, it is possible to move to the next step.

6.7. BRÜCKNER PROFILING

The seventh procedure is devoted to optimizing the volume of the soil to be handled. To do this, the Brückner profile tool is used.

The input file needed is “DiagramArea.txt” and the output file given is “BrucknerProfiling.txt”.

The information is listed in this way:

- The volume of the section and
- The x-coordinate,
- The initial y-coordinate, and
- The final y-coordinate.

The following table is an extract of the output:

Volume (m^3)	X-coordinate	initial Y-coordinate	final Y-coordinate
…	…	…	…
5466.25	270.000	140.000	130.000
5840.1	280.000	140.000	130.000
6216.25	290.000	140.000	1300.000
…	…	…	…

Table 7.9 Extract of the Brückner profiling

The following pictures show the Brückner profiling of the same sections whose area diagrams are shown in pictures 6.42, 6.43, 6.44, 6.45.

Because all the areas are positive, all the volumes are positives. It is not possible to optimize the movements of the soil from the cut sections to the fill sections.

The following pictures show the Bruckner profiling of the same sections provided in the previous subchapter.

The first example is the Brückner profiling of the section between 140 m and 130 m. The final value of the volume is 112727 m^3.

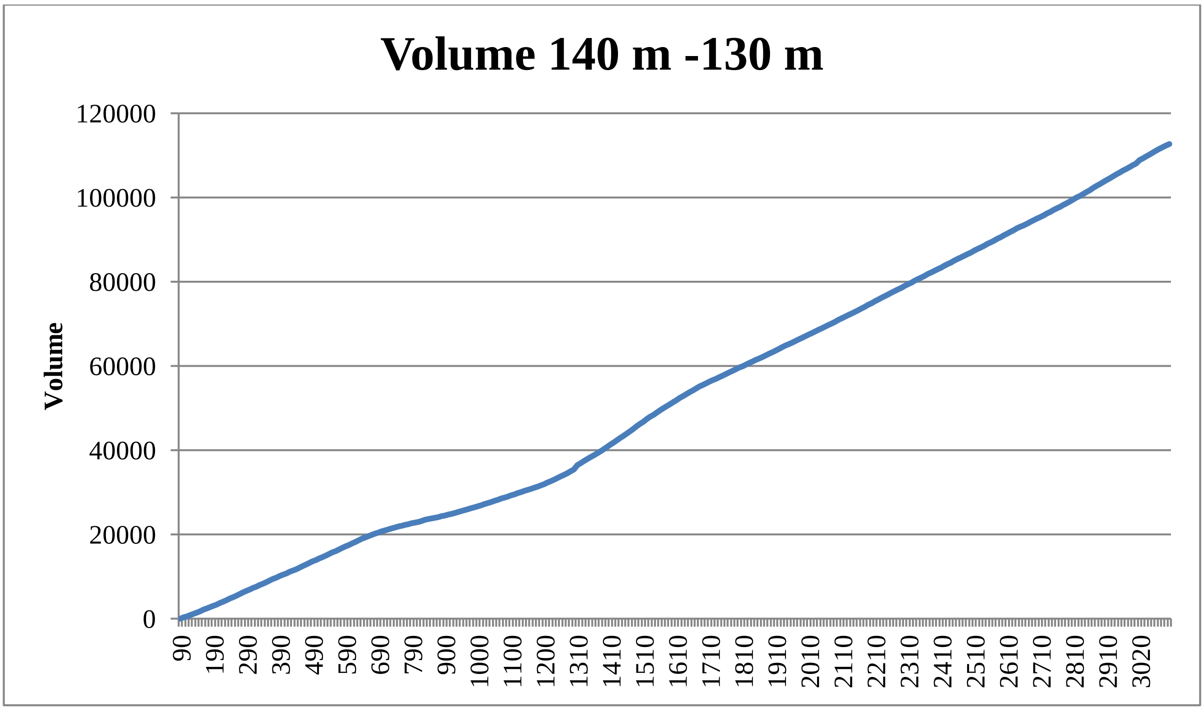


Figure 7.46 Brückner profiling, section #2

The second example is the Brückner profiling of the section between 80 m and 70 m. The final value of the volume is 41624,6 m^3.

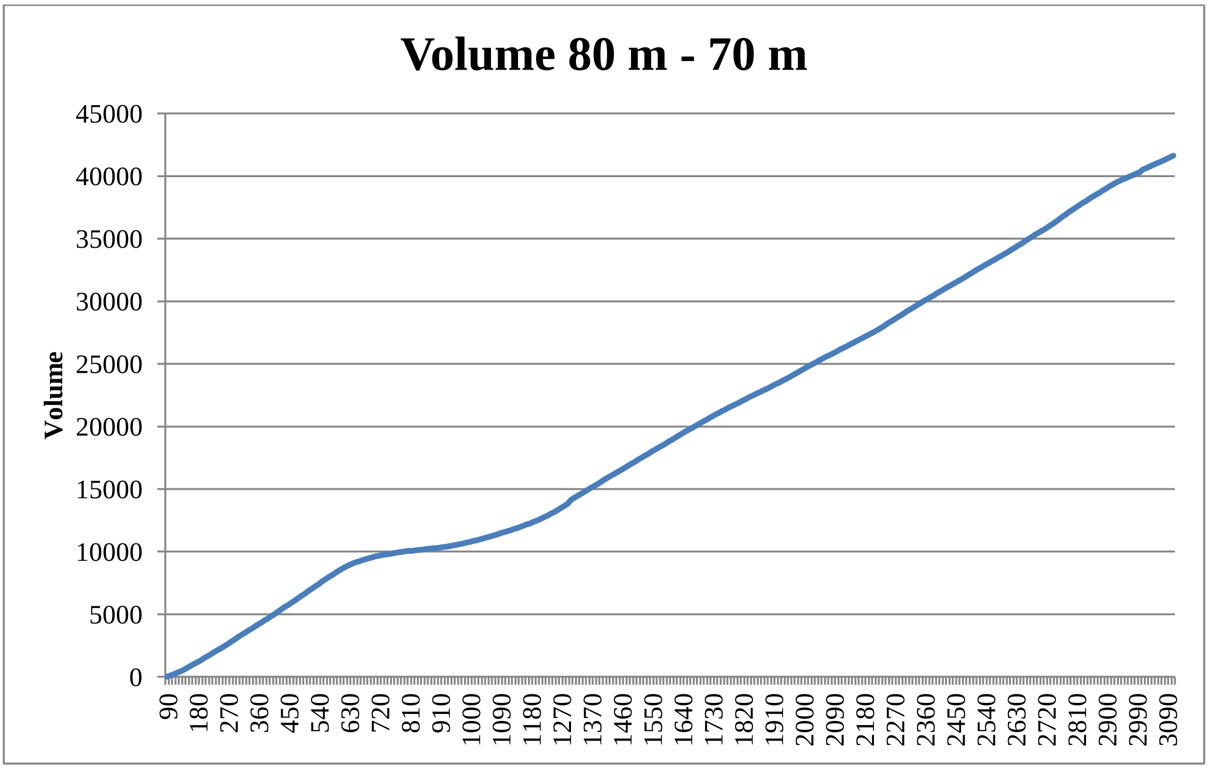


Figure 7.47 Brückner profiling, section #8

The third example is the Brückner profiling of the section between 60 m and 50 m. The final value of the volume is 17890 m^3.

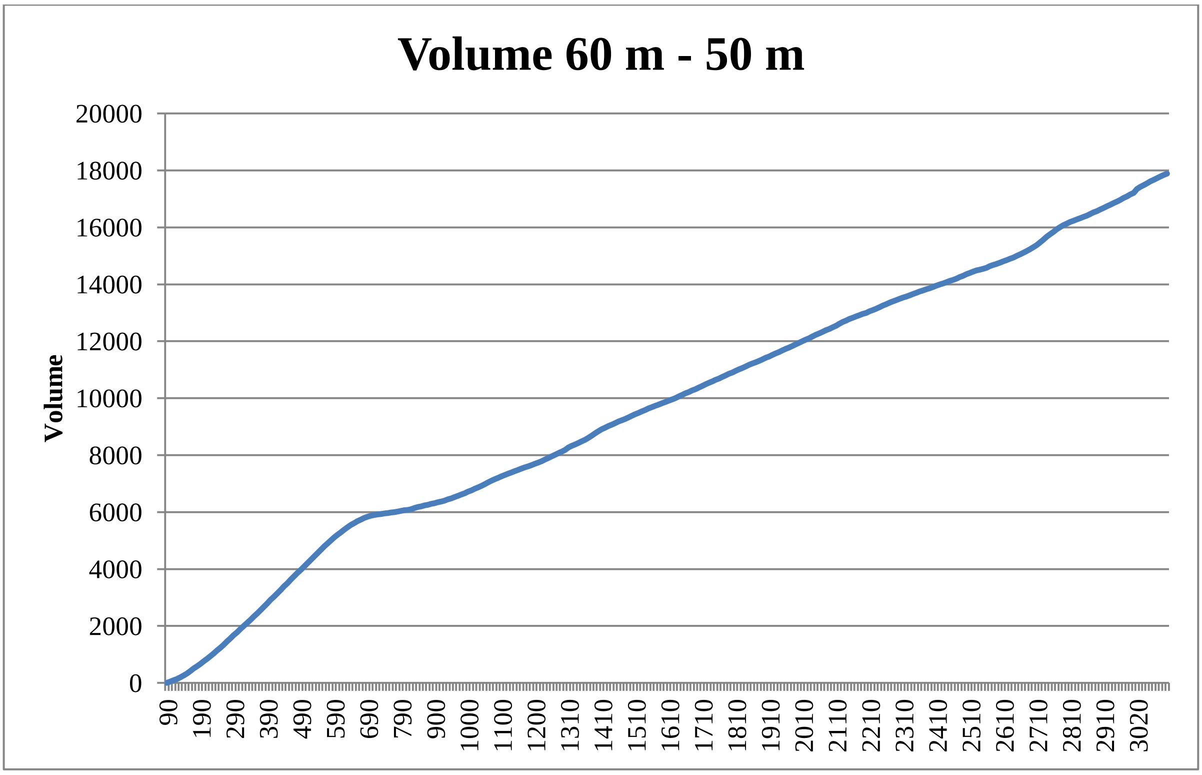


Figure 7.48 Brückner profiling, section #10

The fourth example is the Brückner profiling of the section between 30 m and 20 m. The final value of the volume is 1432,85 m^3.

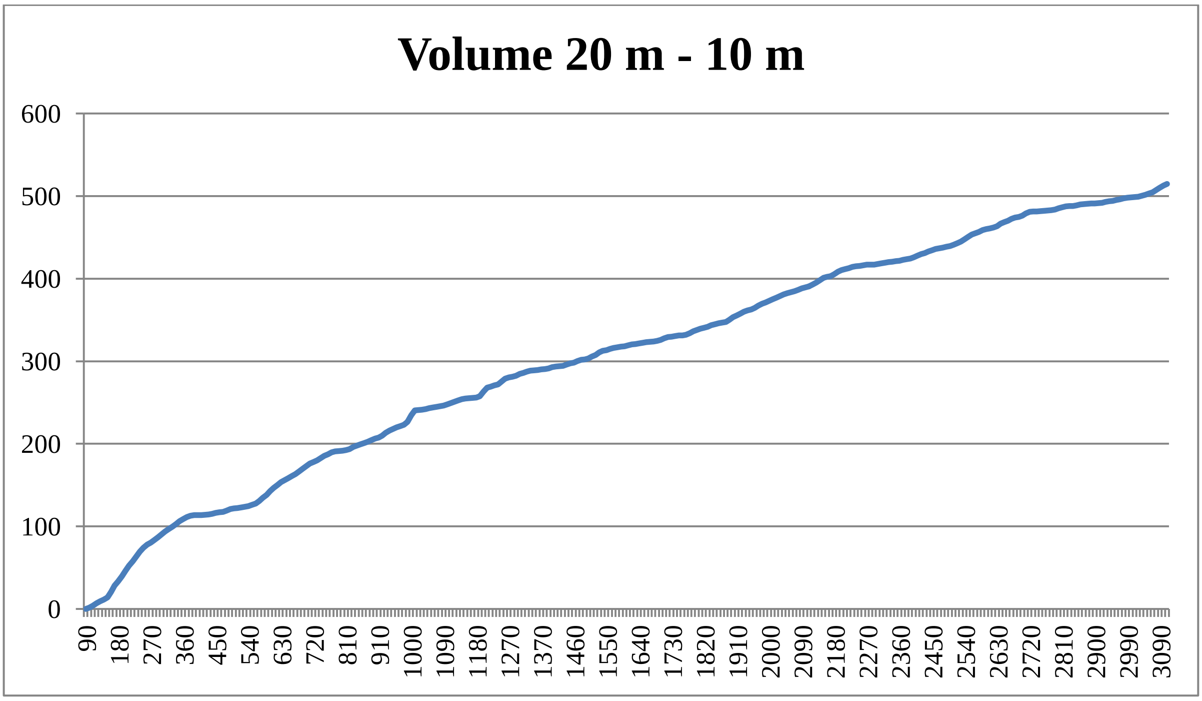


Figure 7.48 Brückner profiling, section #15

CHAPTER 8

The third case study:
the aquifer, Milan. Italy.

The case study is of the aquifer in the city of Milan (Lombardy, north of Italy) and the surrounding area of about 580 km^2.

Until now, the 3D models of the aquifers were generated using interpretative stratigraphic sections or, more recently, the Kirging technique. This last method is stochastic, and its aim is to estimate the value of the field at an unobserved location from a combination of the observations.

Two examples of the application of this method are:

- the paper "Analisi della struttura spaziale di variabili idrogeologiche con tecniche geostatistiche", written by Paola Gattinoni in 2009 and
- the paper "Stima dei volumi di acqua in un acquifero eterogeneo: l'esempio della provincia di Milano", written by T. Bonomi, P. Canepa and F. Del Rosso in 2009.

The method proposed in this dissertation is the Delaunay triangulation and the parametric equation of a straight line passing through two points.

The main difference between the Kirging method and the one presented here is that the former is stochastic, while the latter is a deterministic procedure. The main advantage of the triangulation method is that an input point's estimated value is the same as its actual measured value.

The following subchapters explain in detail the method.

8.1. HYDROGEOLOGICAL SETTING

8.1.1.Introduction

The city of Milan lies on the northern margin of the alluvial plain of the Po River.

The hydrogeological structure of the area has been the object of a number of investigations (Regione Lombardia – ENI, 2002, and references therein).

The water-bearing alluvial deposits are more than 250 m thick and they can be subdivided into four large-scale geological units denoted (from the youngest to the oldest) by A, B, C and D Aquifer Groups. These depositional cycles are separated by clayey deposits which form the Aquitards.

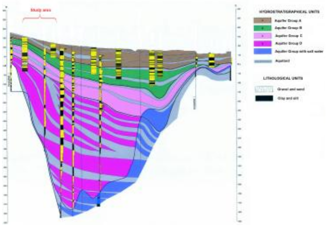


Figure 8.1 N-S hydrogeological cross-section at regional scale (from Regione Lombardia – ENI, 2002, modified)

More in detail, the aquifers traditionally exploited in the study area (A and B) consist of a hydrogeological system of unconfined and semi-confined aquifers within a system of very heterogeneous layers. The groundwater flow occurs form North to South, with a piezometrical gradient ranging between 0.5% in the Northern area and 0.3% in the Southern area.

8.1.2.Field measurements

The data-base developed within the Italian geologic cartography project (CARG) was used to obtain a detailed hydrogeological characterization of the study area. In particular, more than 4,000 stratigraphies were analyzed, generally arising from wells and bore holes which interest mainly the Aquifer Groups A and B, that is the first 100-150 m of depth.

The stratigraphical analysis also allowed to reconstruct the geometry and the spatial trend of the Aquifer Groups A and B. In particular, the bottom surfaces of both Aquifer Group A and B were identified in the stratigraphies, and the obtained data were used for the geostatistical analysis. In particular, variograms showed different structures at different scales of analysis (Figure 8.2):

- at small scale (lag = 0.8 km), the variogram reproduce the local continuity of the clayey layers, even if with a nugget effect is present as a consequence of the parameter variability at a scale smaller than the sample distance,
- at intermediate scale (lag = 3 km), the variogram model is spherical, showing the presence of trend inversion (e.g., at the transition between a paleo-riverbed and an interfluvio structure), and
- at large scale (lag = 12 km), the variogram (fitted with a parabolic model) shows the regional trend of the hydrogeological structure.

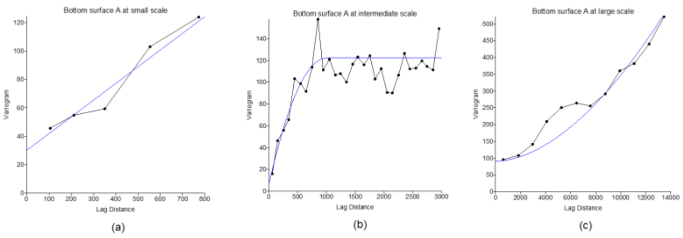


Figure 8.2 Variograms of the bottom surface of the Aquifer Group A at different scales of analysis: (a) small scale; (b) intermediate scale; (c) large scale

In general, the bottom of the Aquifer Group A has a dip direction towards S-SE with altitude of 150 to 50 m a.s.l. (Figure 8.3a). The bottom surface of the Aquifer Group B has the same dip direction, with altitude of 130 to -30 m a.s.l. (Figure 7b).

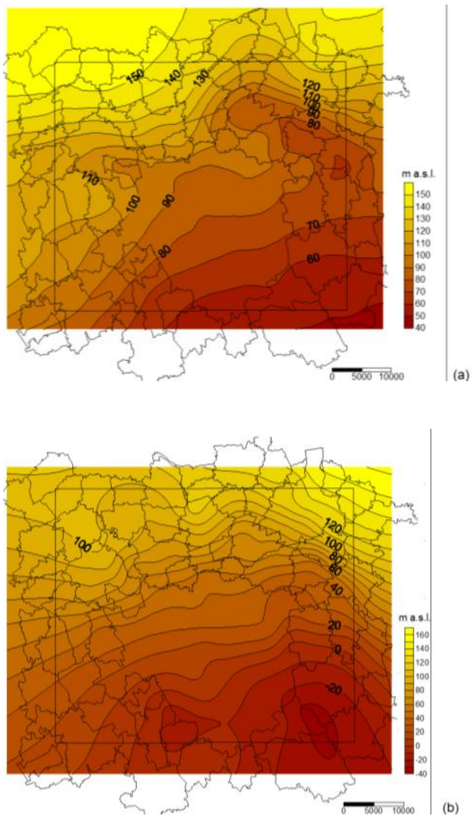


Figure 8.3 Bottom surfaces of the Aquifer Groups A (a) and B (b) in m a.s.l.; the interpolations were made using the large scale estimator (Figure 4) for describing the regional trend.

The Aquitard separating the two aquifer groups is present only in the Southern zone; its thickness is increasing towards S-E, where it reaches 10-12 m with the consequent complete separation of the two aquifers (Figure 8.4).

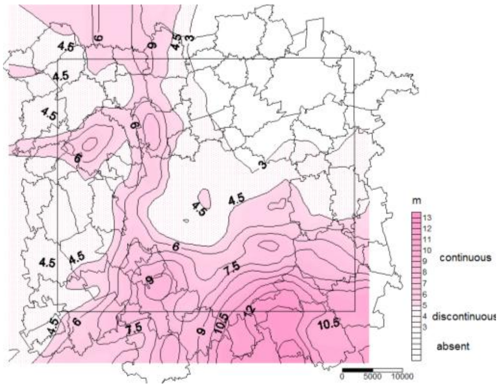


Figure 8.4 Aquitard presence and thickness (in m)

8.1.3.Training images construction

In this study, training images (i.e., conceptual explicit representations of the expected spatial distribution of hydraulic properties or facies types) were constructed based on stratigraphical information of wells and bore holes. These training images were afterwards used to characterize the patterns of geological heterogeneity. The main idea is to borrow geological patterns from these training images and anchor them the subsurface data domain.

At this aim, fifteen two-dimensional vertical training images of gravel, sand and clay occurrence in different orientations were constructed (Figure 8.5), based on filed observations of the geometry and dimensions of the sedimentary structures, by manually interpolating and extrapolating the geological field data.

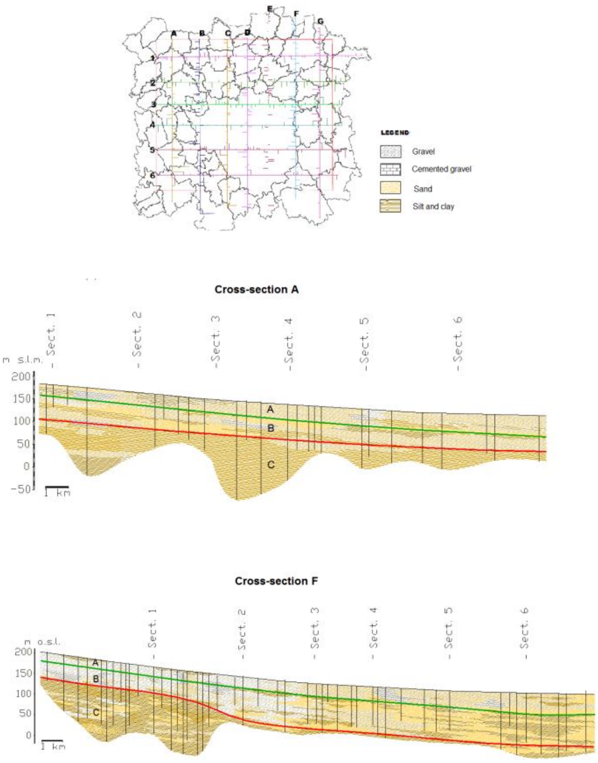


Figure 8.5 Hydrogeological cross-sections network, with two representative N-S sections. The bottom surfaces of Aquifer Groups A and B are shown in the sections.

Generally, the N-S sections, aligned along the groundwater flow direction, show a slight dip of the finest deposits, with a trend characterized by grain size reduction from North to South: sand and clay take the place of gravel, which area are almost absent in the Southern part of the study.

On each training image, a visual analysis was subsequently carried out to obtain a first indication on the connectivity degree; generally, a good level of connectivity is shown, particularly in the central area of the study zone. As example, two cross-sections are reported, representative of structures having respectively high (Section A in Figure 6) and low (Section F in Figure 8.5) connectivity.

8.2. INPUT DATA SET

The aquifer is modeled by points defined by their three coordinates and layer type.

Gauss – Boaga, a transversal conformal cylindrical reference system, is the adopted reference system.

The input file is a list containing ID, North, East, and height values, as shown in the extract below.

Layer “Aquiclude”			
ID	**North**	**East**	**Height (m)**
992	1520090	5030310	-22.139999
993	1520235	5030315	-23.400002
995	1521419	5032874	-14.370003
…	…	…	…

Table 8.1 Extract of the input file

The input data are composed of four files, one for each layer. Each of them contains the coordinates of the points that belong to that specific layer.

More in details, the four layers are:

- land surface,
- Aquitard up,
- Aquitard down, and
- Aquiclude.

The following image shows the input data using four different colors:

- green for the points classified as belonging to the land surface layer,
- red for the points classified as belonging to the Aquitard up layer,
- light blue for the points classified as belonging to the Aquitard down layer, and
- blue for the points classified as belonging to the Aquiclude layer.

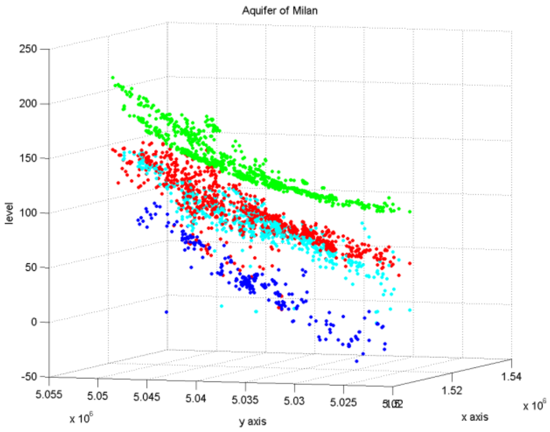


Figure 8.6 3D plot of the input file

Some problems must be solved to obtain a consistent 3D model. The first problem originates from the fact that if the input data contains some outliers, the predicted values will be wrong.

The second problem relates to the fact that the input data is discrete and not equally spaced, uniform or homogeneous. The outputted model must be continuous. This implies that the model is obtained using interpolated techniques.
It is possible to solve these problems subdividing the data processing into six steps.

1. The input data is checked in order to detect and remove the outliers.

2. The information is divided into clusters. This procedure assigns observations into subsets.

3. Each cluster is interpolated using a deterministic procedure called Delaunay triangulation. This method interpolates scattered points into a continuous field.

4. Closed paths between points that belong to a specific layer are built. It is necessary to bind each layer in order to know where to interpolate and where to extrapolate. The adopted method is the following:

   The Delaunay triangulation generates triangles between points. Each triangle is composed of three sides, and if a side is in common with more than one triangle, it is deleted. On the contrary, the remaining sides are classified as a side of the closed path.

5. It is necessary to search for corresponding points that belong to consecutive layers to generate the vertical boundary.
   This procedure is carried out using the relational matching method, as explained in chapter 2, section 5.

6. It is necessary to obtain new information from the Delaunay triangulation. This procedure is carried on using the interpolation method explained in chapter 2, section 7.

Below the various steps are explained in details.

8.3. OUTLIER DETECTION

First of all, outliers in the initial dataset must be detected and removed because the 3D model has to be as reliable as possible. Secondly, triangulation is not a robust procedure: this means that the procedure is not able to find and remove outliers.
For these two reasons, the input data must be analyzed and, if it is necessary, corrected.
In this specific case there are two kinds of outliers:

1. Points exist that are duplicates: they have the same planimetric coordinates, but different IDs and heights.
2. Additionally, some points neither belong to the layer they are classified in nor to the point cloud.

Starting from the first kind of outlier, duplicate points are characterized by the same coordinates, but different ID and height.
The input files needed are:

- “LSInput.txt”,
- “ADInput.txt”,
- “AUInput.txt”, and
- “AQInput.txt”.

The output files given are:

- “LSInputRevI.txt”,
- “ADInputRevI.txt”,
- “AUInputRevI.txt”, and
- “AQInputRevI.txt”.

The 3D output plots given are:

- “LSInputRevI.tif”,
- “ADInputRevI.tif”,
- “AUInputRevI.tif”, and
- “AQInputRevI.tif”.

No duplicate points are found in the land surface layer so the file “LSInput.txt” is equal to “LSInputRevI.txt”. The following two images show the land surface file before and after the outlier detection.
Figure 8.7 shows the land surface layer set that will be processed.

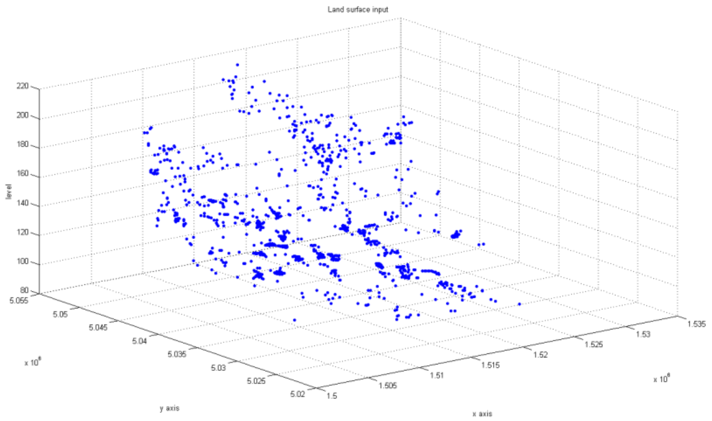


Figure 8.7 Land surface layer

Eight duplicate points are found in the Aquitard up layer. The following table provides their coordinates.

Layer “Aquitard up”			
ID	**North**	**East**	**Height**
3079	1521420	5032457	60.8200
3195	1521420	5032457	73.8200
3086	1521499	5032573	61.1000
3202	1521499	5032573	74.1000
3088	1521419	5032874	61.6300
3175	1521419	5032874	71.6300
3098	1521301	5032595	63.0200
3112	1521050	5032900	64.6700
3238	1521050	5032900	76.6700
3133	1520951	5033023	66.6800
3219	1520951	5033023	75.6800

Table 7.2 Double points in the Aquitard up

In Figure 8.8 the double points are shown in red, the others in blue.

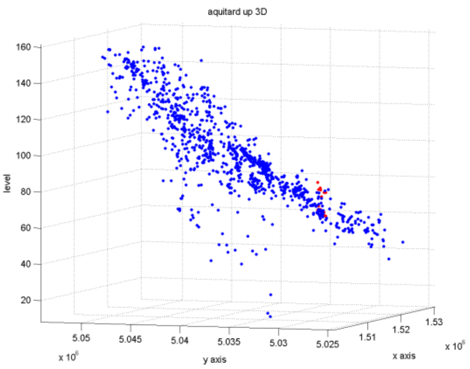


Figure 8.8 Points (in blue) and duplicate points (in red) in the Aquitard up

The objective at this point is to decide which points must be deleted and which one must be used. The determining method is explained in chapter 2, section 1.
The image 7.9 shows the only points that will be processed.

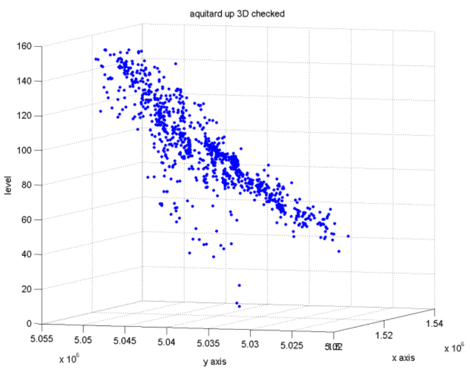


Figure 8.9 Checked points in the Aquitard up

No duplicate points are found in the Aquitard down layer.
 Figure 8.10 shows the Aquitard down layer data set.

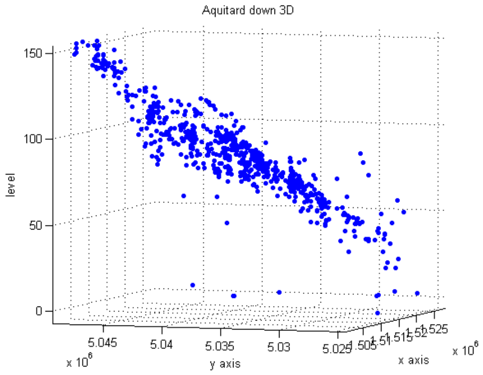


Figure 8.10 Aquitard down layer data set

No duplicate points are found in the Aquiclude layer.
Figure 8.11 displays the Aquiclude layer.

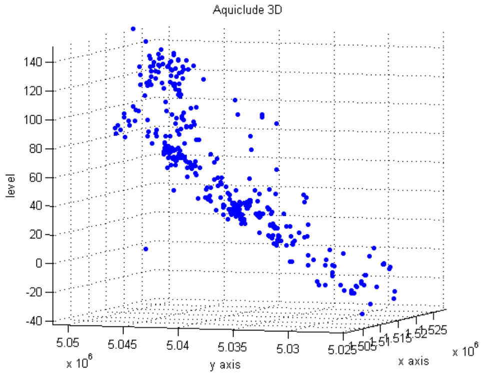


Figure 8.11 Aquiclude layer

The second type of outliers is the following: some points are classified as belonging to a layer, but they do not belong to that layer. Those points are located either outside the point cloud or in a wrong position.
The input files needed are:

- “LSInputRevI.txt”,
- “ADInputRevI.txt”,
- “AUInputRevI.txt”, and
- “AQInputRevI.txt”.

The output files given are:

- “LSInputRevII.txt”,
- “ADInputRevII.txt”,

- “AUInputRevII.txt”, and
- “AQInputRevII.txt”.

The 3D output plots given are:

- “LSInputRevII.tif”,
- “ADInputRevII.tif”,
- “AUInputRevII.tif”, and
- “AQInputRevII.tif”.

No incorrect points are found in the land surface layer. For this reason, the file “LSInputRevI.txt” is equivalent to the file named “LSInputRevII.txt”. The following image shows the points belonging to Aquitard down layer. The points in red are kept; the ones in blue are deleted.

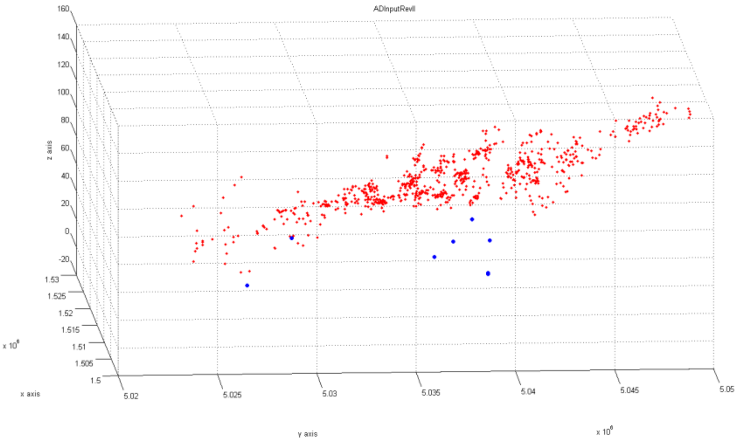


Figure 8.12 Points deleted from the Aquitard down

The following table provides the contents of the “ADInputRevII.txt” file, along with the eight IDs and coordinates of the deleted points from the Aquitard down layer.

Layer “Aquitard down”			
ID	**East**	**North**	**Height**
2001	1520090	5030160	28.650001
2093	1520944	5027966	-7
2155	1525405	5037712	1
2233	1510198	5037587	49
2259	1510867	5038597	63.300003
2401	1520135	5040035	1
2402	1520515	5040074	1
2622	1502570	5038900	67.899994

Table 7.3 Points deleted from the Aquitard down

The following image shows the points belonging to Aquitard up layer. The points in red are kept; the ones in blue are deleted.

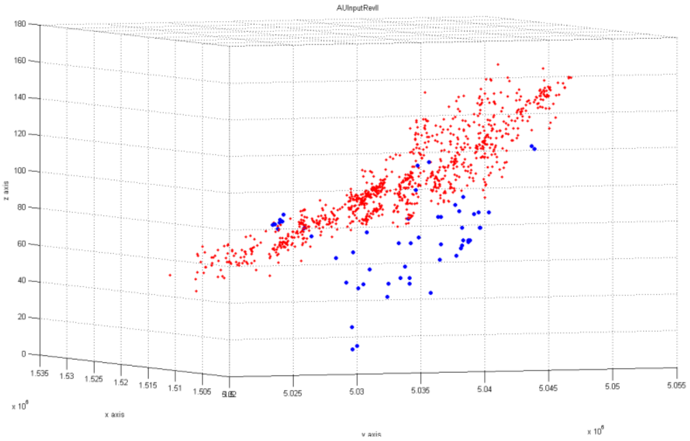


Figure 8.13 Points deleted from the Aquitard up

The following table provides the contents of the "AUInputRevII.txt" file and the IDs and the three coordinates of the deleted points from the Aquitard up layer.

Layer "Aquitard up"			
ID	**East**	**North**	**Height**
3000	1513687	5035457	8.19
3001	1513349	5035631	10.09
3002	1514113	5035555	20.09
3003	1516721	5039436	35.30
3004	1524994	5040701	37.00
3005	1512624	5041114	38.60
3006	1524856	5041040	39.00
3007	1525341	5039905	39.90
3008	1522950	5042170	40.00
3010	1522265	5042814	43.00
3011	1522590	5043680	43.00
3012	1512023	5039191	43.90
3016	1524950	5041559	47.00
3020	1522720	5043369	49.00
3030	1526648	5039632	53.00
3037	1519410	5044683	53.80
3059	1521027	5038613	58.20
3081	1506024	5040307	61.00
3090	1522470	5043710	62.00
3095	1518409	5044351	62.50
3109	1515908	5045422	64.40
3115	1506271	5040781	65.00
3116	1522283	5044270	65.00
3117	1516391	5045769	65.10
3120	1515841	5045397	65.40
3129	1506347	5040909	66.50

3138	1510198	5037587	67.00
3139	1525930	5041745	67.00
3155	1506459	5041029	69.30
3156	1526156	5036918	69.70
3175	1521419	5032874	71.63
3176	1517654	5045694	71.70
3177	1508919	5030212	72.00
3195	1521420	5032457	73.82
3199	1521301	5032595	74.02
3202	1521499	5032573	74.10
3217	1521182	5032903	75.24
3219	1520951	5033023	75.68
3238	1521050	5032900	76.67
3245	1503190	5040911	77.00
3246	1503261	5041002	77.00
3264	1517912	5044148	77.70
3267	1517462	5043730	78.00
3288	1520884	5033105	79.61
3290	1511061	5038716	79.80
3299	1516313	5047206	80.30
3301	1518920	5046016	80.30
3302	1516488	5046491	80.40
3327	1520840	5046490	83.00
3344	1502734	5040309	84.70
3387	1518992	5046367	87.80
3403	1527400	5046219	88.80
3620	1528555	5046824	101.70
3658	1528100	5047540	103.80
3754	1516142	5050727	114.50
3767	1516430	5050610	116.00

Table 7.4 Points deleted from the Aquitard up layer

The following image shows the points belonging to Aquiclude layer. The points in red are kept; the ones in blue are deleted.

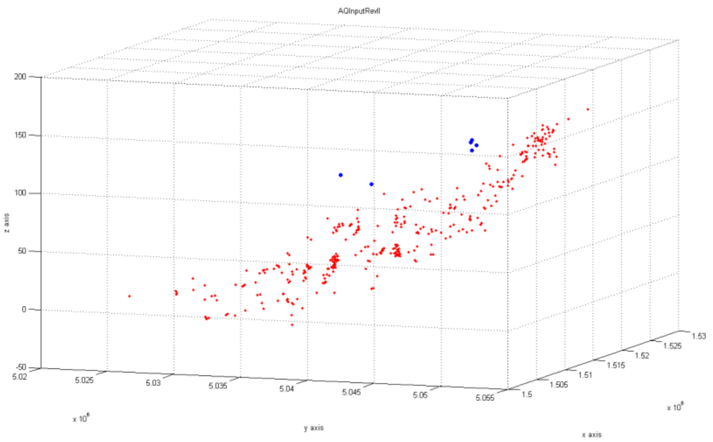


Figure 8.14 Points deleted from the Aquiclude

The following table provides the contents of the "AQInputRevII.txt" file and the IDs and the three coordinates of the deleted points from the Aquiclude layer.

Layer "Aquiclude"			
ID	**East**	**North**	**Height**
1259	1511750	5047316	131.5
1260	1512859	5046810	138.5
1261	1512877	5046733	136.5
1262	1514454	5046505	131.199997
1273	1502939	5041209	122
1282	1504840	5042713	112

Table 7.5 Points deleted from the Aquiclude layer

The first step is completed. It is now possible to move to the next procedure.

8.4. CLUSTERS ANALYSIS

The second procedure is focused on the clusters analysis. As explained in chapter 2, section 3, clustering means the operation that assigns a set of observations to subsets called clusters.

As mentioned before, the aquifer of Milan is composed of four layers. For this reason, the clusters of this 3D model are the four layers that define the aquifer.

The input files needed are:

- "LSInputRevIII.txt",
- "ADInputRevIII.txt",
- "AUInputRevIII.txt", and
- "AQInputRevIII.txt".

The output files given are:

- “LSCluster.txt”,
- “ADCluster.txt”,
- “AUCluster.txt”, and
- “AQCluster.txt”.

The 3D output plots given are:

- “LSCluster.tif”,
- “ADCluster.tif”,
- “AUCluster.tif”, and
- “AQCluster.tif”.

Figure 8.18 shows the land surface cluster.

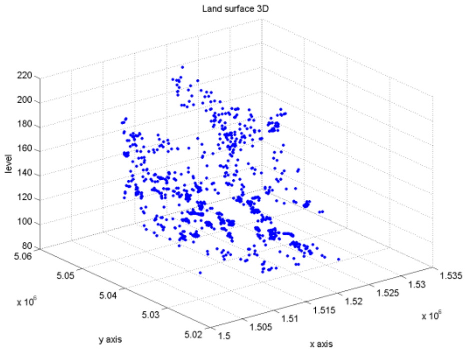


Figure 8.15 Land surface cluster

Figure 8.16 shows the Aquitard up cluster.

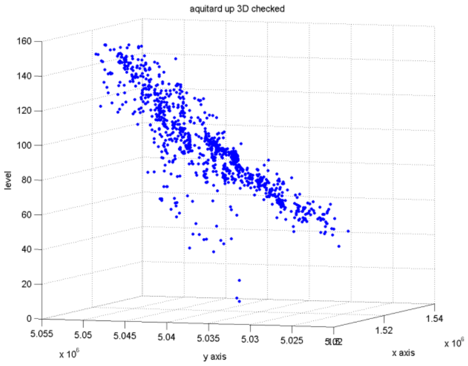


Figure 8.16 3D Aquitard up cluster

Figure 8.17 shows the Aquitard down cluster.

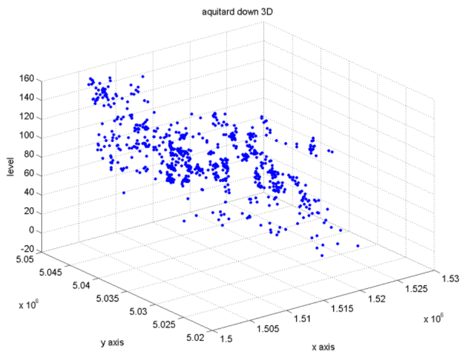


Figure 8.17 Aquitard down cluster

Figure 8.18 shows the Aquiclude cluster.

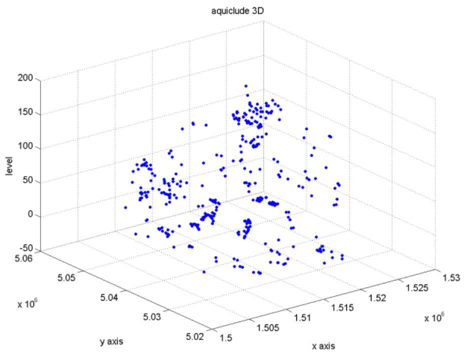


Figure 8.18 Aquiclude cluster

The following image shows the data with different colors, one for each cluster.

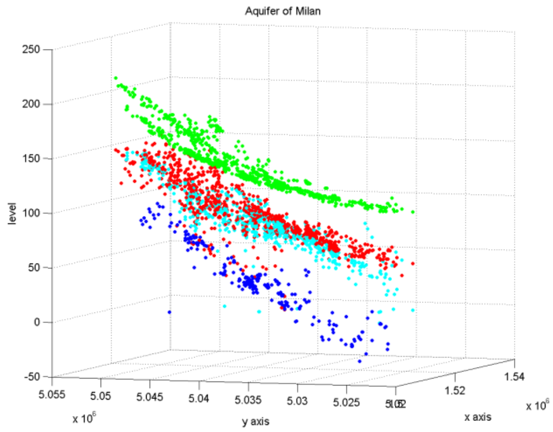


Figure 8.19 3D plot of the input file

The second procedure is completed. It is now possible to move to the next step.

8.5. DELAUNAY TRIANGULATION

After the cluster procedure, the data is interpolated. The adopted method is the Delaunay triangulation and its properties are described in chapter 2, section 6.
The input files needed are:

- “LSCluster.txt”,
- “ADCluster.txt”,
- “AUCluster.txt”, and
- “AQCluster.txt”.

The output files given are:

- “LSTriangulation.txt”,
- “ADTriangulation.txt”,
- “AUTriangulation.txt”, and
- “AQTriangulation.txt”.

The 3D output plots given are:

- “LSTriangulation.tif”,
- “ADTriangulation.tif”,
- “AUTriangulation.tif”, and
- “AQTriangulation.tif”.

The following four images show the Delaunay triangulation of the four layers. The data acquired are in black and the Delaunay triangulation assumes different colors, depending on the height of the points.

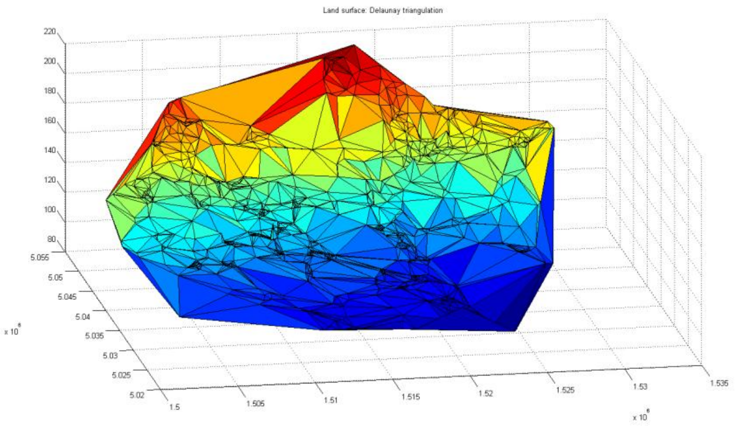


Figure 8.20 Triangulation of land surface cluster

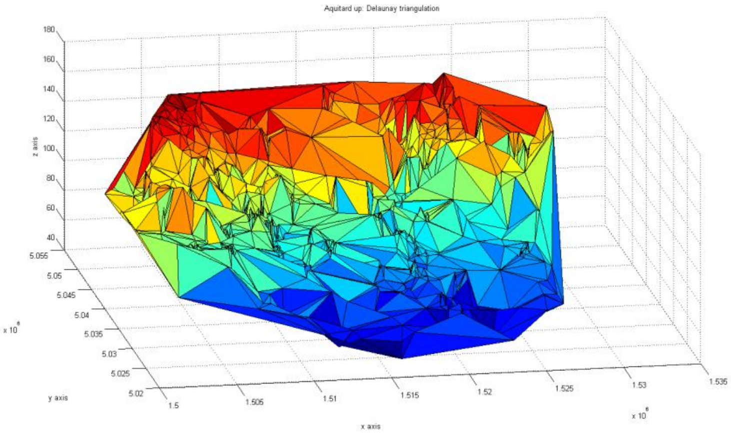


Figure 8.21 Triangulation of Aquitard up cluster

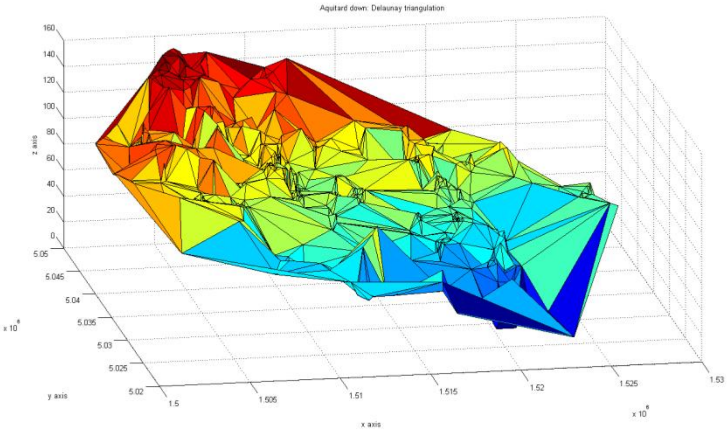


Figure 8.22 Triangulation of Aquitard down cluster

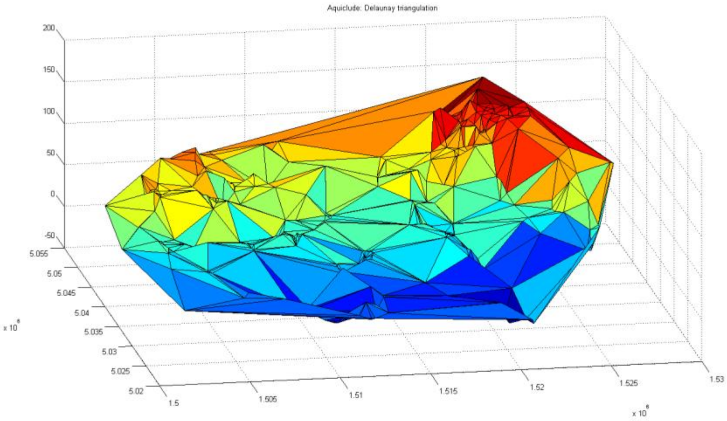


Figure 8.23 Triangulation of Aquiclude cluster

After the Delaunay triangulation, each cluster is composed of a continuous layer and is no longer discrete. This step is completed and it is now possible to move to the next one.

8.6. FEATURE EXTRACTION

After the Delaunay triangulation, it is necessary to extract the boundary of the model. This is necessary to determine if points are to be interpolated or extrapolated, based on whether they fall inside or outside the boundary, respectively. For this reason, closed paths between points that belong to a specific cluster are built.
The input files needed are:

- "LSTriangulation.txt",
- "ADTriangulation.txt",
- "AUTriangulation.txt", and
- "AQTriangulation.txt".

The output files given are:

- "LSPerimeter.txt",
- "ADPerimeter.txt",
- "AUPerimeter.txt", and
- "AQPerimeter.txt".

The 3D output plots given are:

- "LSPerimeter.tif",
- "ADPerimeter.tif",
- "AUPerimeter.tif", and
- "AQPerimeter.tif".

Figure 8.24 shows the land surface perimeter in blue, the points classified as perimeter in red and the point ID in black.

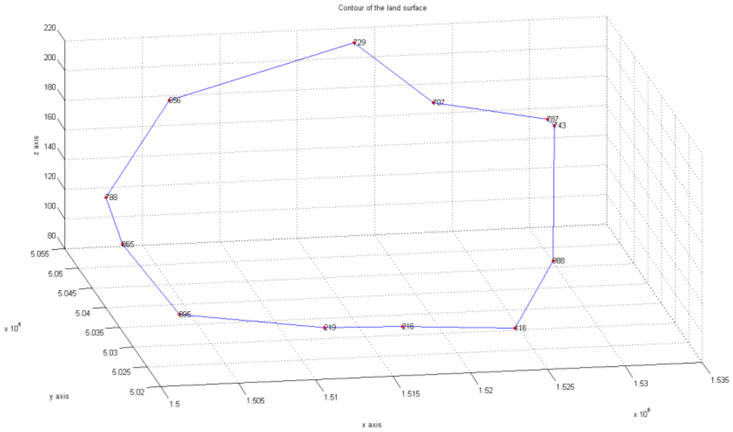


Figure 8.24 Contour of land surface cluster

The following table provides the contents of the file named "LSPerimeter.txt" and the IDs / three coordinates of the points classified as belonging to the perimeter.

“Land surface” perimeter			
ID	**East**	**North**	**Height**
707	1523231	5051742	176.10
729	1518512	5054102	212.70
656	1505680	5049150	193.00
788	1500031	5040238	153.70
865	1500221	5035303	135.40
895	1502239	5025712	112.60
219	1511429	5024772	101.90
216	1516404	5024361	101.50
116	1523874	5025488	93.50
988	1528083	5035504	110.40
743	1530269	5047454	168.90
787	1529930	5047980	172.00
707	1523231	5051742	176.10

Table 7.6 Perimeter: land surface

Figure 8.25 shows the Aquitard up perimeter in blue, the points classified in red and the point ID in black.

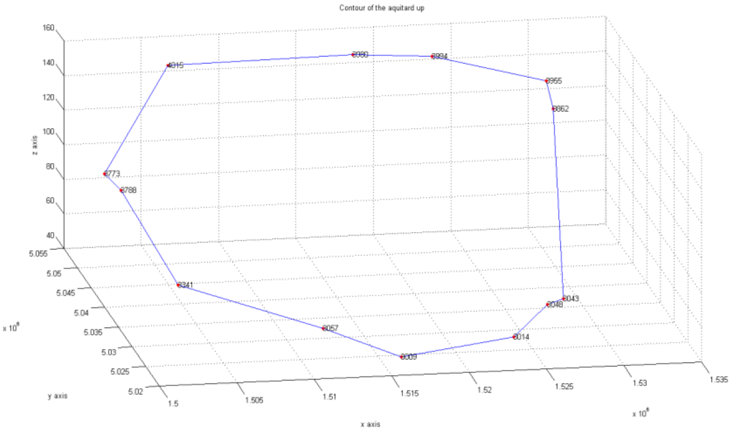


Figure7.25 Contour of the Aquitard up cluster

The following table provides the contents of the file named “AUPerimeter.txt” and the IDs / three coordinates of the points classified as belonging to the perimeter.

“Aquitard up” perimeter			
ID	**East**	**North**	**Height**
3341	1502239	5025712	84.60
3378	1500221	5035303	118.40
3773	1500031	5040238	116.70
4015	1505680	5049150	157.00
3980	1518512	5054102	146.70
3994	1523231	5051742	149.10

3995	1529930	5047980	141.00
3862	1530269	5047454	125.90
3043	1528010	5030770	55.00
3048	1526610	5028765	56.70
3014	1523874	5025488	46.50
3009	1516404	5024361	40.50
3057	1511429	5024772	57.90
3341	1502239	5025712	84.60

Table 7.7 Perimeter: Aquitard up

Figure 8.26 shows the Aquitard down perimeter in blue, the point in red and the point ID in black.

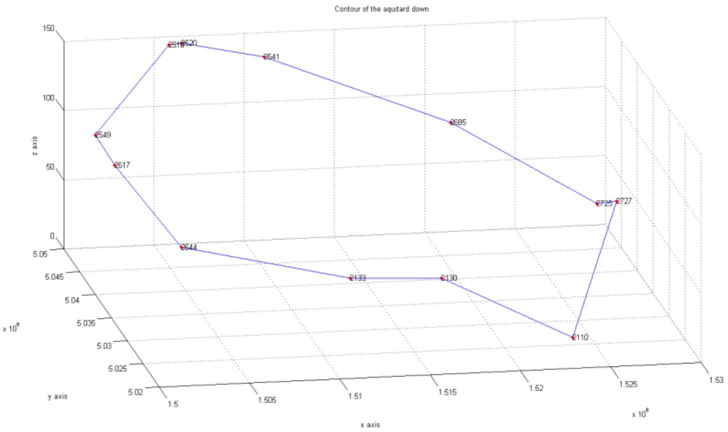


Figure 8.26 Contour of the Aquitard down cluster

The following table provides the contents of the file named “ADPerimeter.txt” and the IDs / three coordinates of the points classified as belonging to the perimeter.

“Aquitard down” perimeter			
ID	**East**	**North**	**Height**
2510	1505680	5049150	147.00
2549	1500031	5040238	114.70
2617	1500221	5035303	109.40
2644	1502239	5025712	80.60
2133	1511429	5024772	55.90
2130	1516404	5024361	54.50
2110	1523874	5025488	3.50
2727	1528083	5035504	66.40
2725	1527306	5037339	59.00
2685	1520439	5044385	98.20
2541	1510818	5048391	138.00
2520	1506400	5049266	147.50
2510	1505680	5049150	147.00

Table 7.8 Perimeter: Aquitard down

Figure 8.27 shows the Aquiclude perimeter in blue, the point classified in red and the point ID in black.

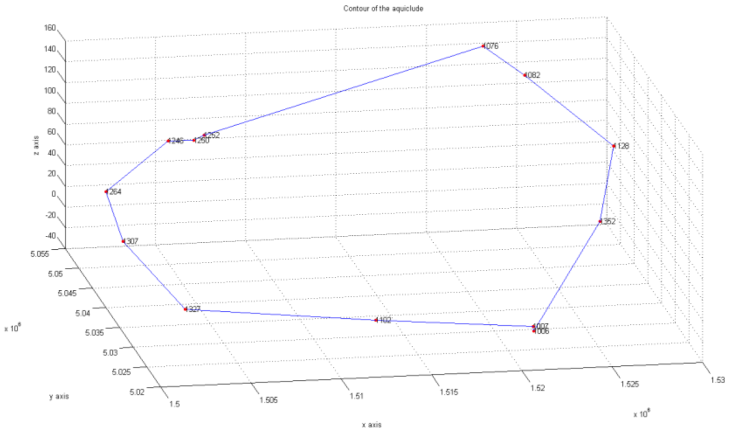


Figure 8.27 Contour of Aquiclude cluster

The following table provides the contents of the file named “AQPerimeter.txt” and the IDs / three coordinates of the points classified as belonging to the perimeter.

“Aquiclude” perimeter			
ID	**East**	**North**	**Height**
1006.00	1521795.00	5027419.00	-31.60
1007.00	1521731.00	5027350.00	-26.60
1102.00	1512640.00	5024830.00	-4.20
1327.00	1502239.00	5025712.00	11.20
1307.00	1500221.00	5035303.00	41.40
1264.00	1500031.00	5040238.00	70.70
1246.00	1504315.00	5045856.00	95.00
1250.00	1506008.00	5047612.00	88.00
1252.00	1506619.00	5047916.00	91.20
1076.00	1522606.00	5051372.00	151.00
1082.00	1524443.00	5048280.00	133.70
1128.00	1528340.00	5041530.00	88.00
1352.00	1526886.00	5037022.00	33.80
1006.00	1521795.00	5027419.00	-31.60

Table 7.9 Perimeter: Aquiclude

The fourth step is completed and it is possible to move to the next one.

8.7. RELATIONAL MATCHING

Fifthly, it is necessary to search for corresponding points that belong to consecutive layers to generate the vertical boundary.

This procedure is carried on using the relational matching method, as explained in chapter 2, section 5.

Relational matching involves all the points classified as belonging to the perimeter. In particular, the land surface perimeter contains twelve points, the Aquitard up perimeter contains thirteen points, the Aquitard down perimeter contains thirteen points, and finally the Aquiclude perimeter contains twelve points.

Relational matching connects the points belonging to one perimeter (i.e. Aquitard up) to the points of the previous one (i.e. land surface) and the following one (i.e. Aquitard down).

The following subchapters describe the relational matching of each pair. The software provides six tables and six 3D plots, one for each of the following pair:

- land surface – Aquitard up,
- Aquitard up – Aquitard down,
- Aquitard down – Aquiclude,
- Aquitard up - land surface,
- Aquitard down - Aquitard up, and
- Aquiclude - Aquitard down.

7.7.1.Land surface and Aquitard up

The input files needed are:

- “LSperimeter.txt” and
- “AUperimeter.txt”.

The output file given is “RelationalLand-Up.txt”. The 3D output plot given is “Relational matching: land surface and Aquitard up.tif”.

The following table provides the information about the relational matching between the points classified as land surface perimeter (first column), the points classified as Aquitard up perimeter (second column), and the distances (in meter) between the matched points.

Relational Matching: land surface and Aquitard up		
Land surface ID	**Aquitard up ID**	**Distance (m)**
707	3994	27
729	3980	66
656	4015	36
788	3773	37
865	3788	17
895	3341	28
219	3057	44
216	3009	61
116	3014	47

988	3043	4734.88692156423
743	3862	43
787	3955	31

Table 7.10 Relational Matching: land surface and Aquitard up

Figure 8.28 shows the 12 red points belonging to the land surface perimeter, the 12 blue points belonging to the Aquitard up perimeter, and the connections between the different points in blue. The numbers in black represent the ID of each point.

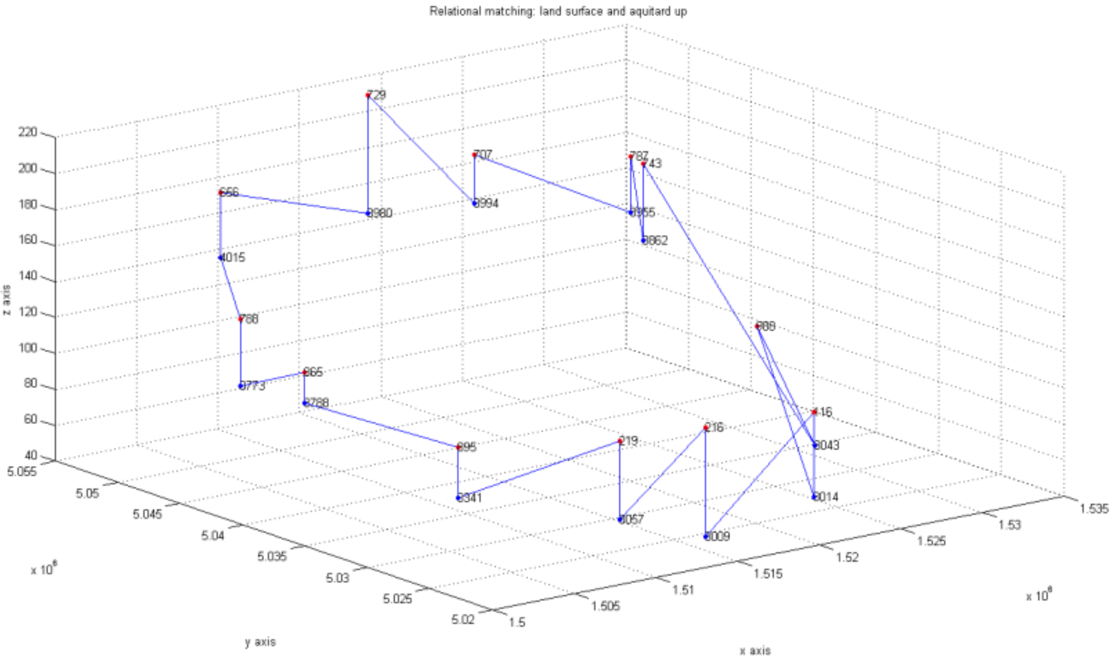


Figure 8.28 Relational matching: land surface and Aquitard up

A check is done on the longest side (4734.88692156423 m). The following image shows the correctness of the longest side.

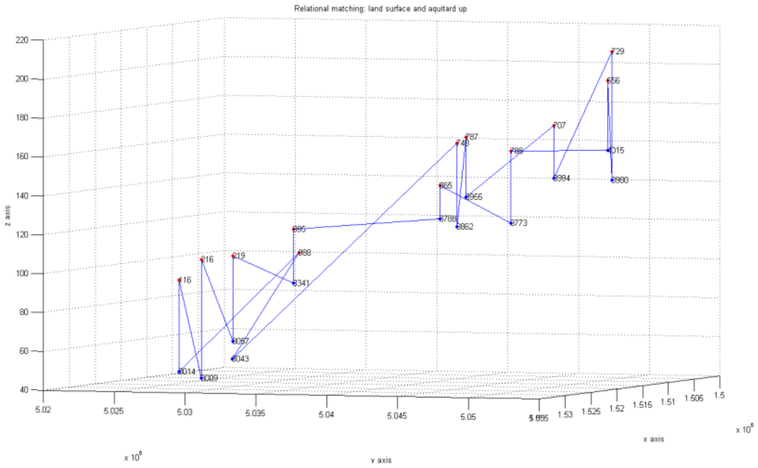


Figure 8.29 Check on the relational matching: land surface and Aquitard up

7.7.2.Aquitard up and Aquitard down

The input files needed are:

- “AUperimeter.txt” and
- “ADperimeter.txt”.

The output file given is “RelationalUp_Down.txt”. The 3D output plot given is “Relational matching: Aquitard up and Aquitard down.tif”.

The following table provides the information about the relational matching between the points classified as Aquitard up perimeter (first column), the points classified as Aquitard down perimeter (second column) and the distances between the matched points.

Relational matching: Aquitard up and Aquitard down		
Aquitard up ID	**Aquitard down ID**	**Distance (m)**
3341	2644	4
3788	2617	9
3773	2549	2
4015	2510	10
3980	2541	9581.92218137885
3994	2685	7869.13615398793
3955	2685	10149.1348320928
3862	2685	10297.9817580922
3043	2727	4734.57653439038
3048	2110	4269.33896991092
3014	2110	43
3009	2130	14
3057	2133	2

Table 7.11 Relational matching: Aquitard up and Aquitard down

Figure 8.30 shows the 13 red points belonging to the Aquitard up perimeter layer, the 12 blue points belonging to the Aquitard down perimeter and the connections between the different points in blue. The numbers in black represent the ID of each point.

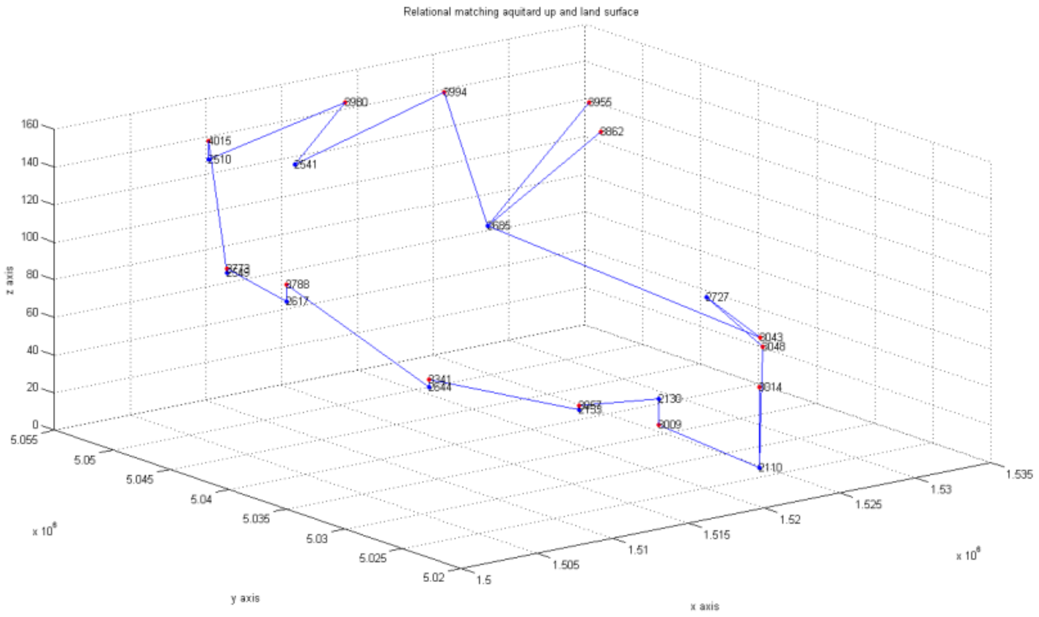


Figure 8.30 Relational matching: Aquitard up and Aquitard down

7.7.3.Aquitard down and Aquiclude

The input files needed are:

- “ADperimeter.txt” and
- “AAperimeter.txt”.

The output file given is “RelationalDown_Aqui.txt”.
The 3D output plot given is “Relational matching: Aquitard down and Aquiclude.tif”.

The following table provides the information about the relational matching between the points classified as Aquitard down perimeter (first column), the points classified as Aquiclude perimeter (second column), and the distances between the matched points.

Relational matching: Aquitard down and Aquiclude		
Aquitard down ID	**Aquiclude ID**	**Distance (m)**
2510	1252	1551.64127297517
2549	1264	44
2617	1307	68
2644	1327	69.4
2133	1102	1213.87685124975
2130	1102	3793.56068753355
2110	1006	2837.64585704418
2727	1352	1933.441429162
2725	1352	526.805504906697
2685	1082	5586.08102787634
2541	1252	4226.04025536908

2520	1252	1368.80630112518

Table 7.12 Relational matching: Aquitard down and Aquiclude

Figure 8.31 shows the 12 red points belonging to the Aquitard down perimeter, the 12 blue points belonging to the Aquiclude perimeter, and the connections between the different points in blue. The numbers in black represent the ID of each point.

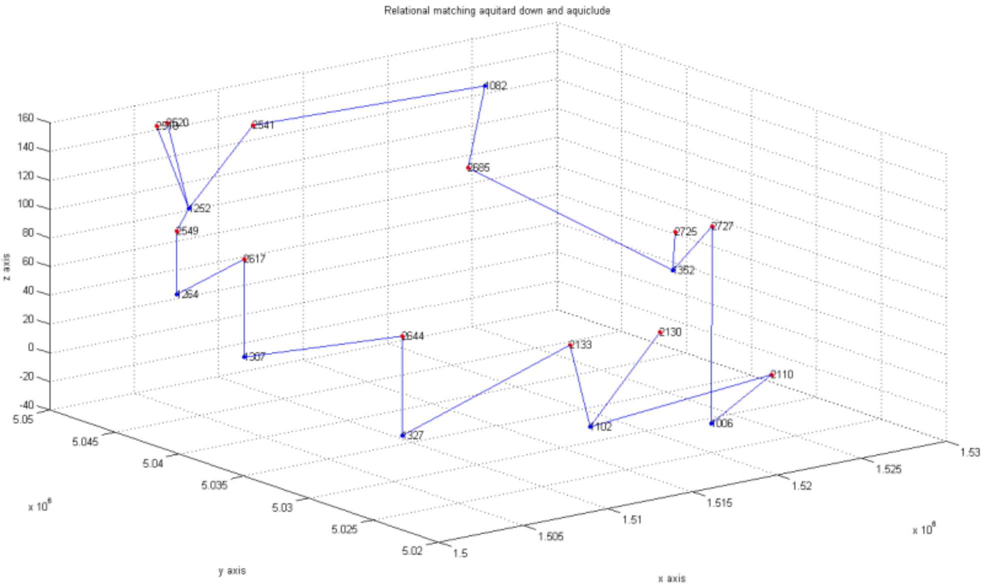


Figure 8.31 Relational matching: Aquitard down and Aquiclude

7.7.4.Aquitard up and land surface

The input files needed are:

- “AUperimeter.txt” and
- “LSperimeter.txt”.

The output file given is “RelationalUp_Land.txt”.
The 3D output plot given is “Relational matching: Aquitard up and land surface.tif”.

The following table provides the information about the relational matching between the points classified as Aquitard up perimeter (first column), the points classified as land surface perimeter (second column) and the distances between the matched points.

Relational matching: Aquitard up and land surface		
Aquitard up ID	**Land surface ID**	**Distance (m)**
3341	895	28
3788	865	17
3773	788	37
4015	656	36
3980	729	66

3994	707	27
3955	787	31
3862	743	43
3043	988	4734.88692156423
3048	116	4269.16610592748
3014	116	47
3009	216	61
3057	219	44

Table 7.13 Relational matching: Aquitard up and land surface

Figure 8.32 shows the xx red points belonging to the Aquitard up perimeter layer, the xx blue points belonging to the land surface perimeter and the connections between the different points in blue. The numbers in black represent the ID of each point.

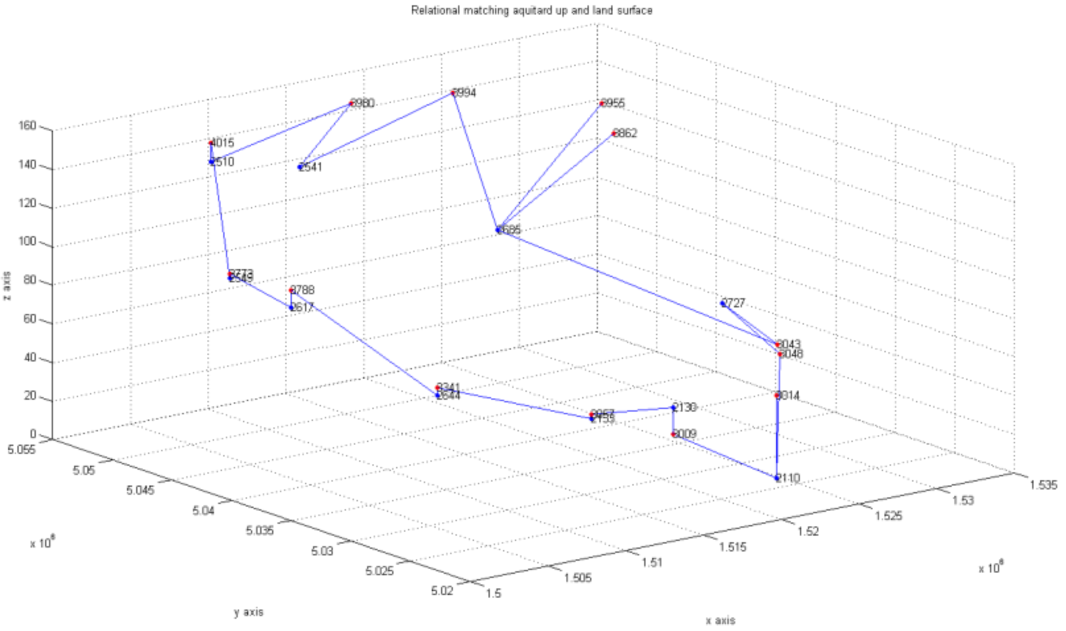


Figure 8.32 Relational matching: Aquitard up and land surface

7.7.5.Aquitard down and Aquitard up

The input files needed are:

- “ADperimeter.txt” and
- “AUperimeter.txt”.

The output file given is “RelationalDown_Up.txt”.
The 3D output plot given is “Relational matching: Aquitard down and Aquitard up.tif”.

The following table provides the information about the relational matching between the points classified as Aquitard down perimeter (first column), the points classified as Aquitard up perimeter (second column), and the distances between the matched points.

Relational matching: Aquitard down and Aquitard up		
Aquitard down ID	**Aquitard up ID**	**Distance (m)**
2510	4015	10
2549	3773	2
2617	3788	9
2644	3341	4
2133	3057	2
2130	3009	14
2110	3014	43
2727	3043	4734.57653439038
2725	3043	6606.61736443091
2685	3994	7869.13615398793
2541	4015	5193.79302629591
2520	4015	729.346454025794

Table 7.14 Relational matching: Aquitard down and Aquitard up

Figure 8.33 shows the 12 red points belonging to the Aquitard down perimeter, the 13 blue points belonging to the Aquitard up perimeter, and the connections between the different points in blue. The numbers in black represent the ID of each point.

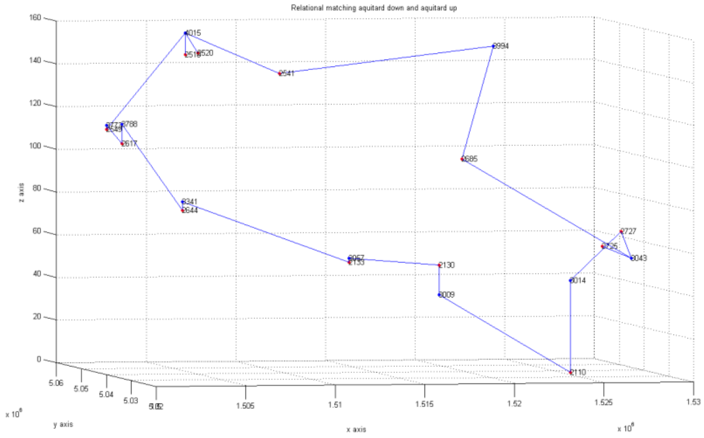


Figure 8.33 Relational matching: Aquitard down and Aquitard up

7.7.6.Aquiclude and Aquitard down

The input files needed are:

- “AAperimeter.txt” and
- “ADperimeter.txt”.

The output file given is “RelationalAqui_Down.txt”.
The 3D output plot given is “Relational matching: Aquiclude and Aquitard down”.

The following table provides the information about the relational matching between the points classified as Aquiclude perimeter (first column), the points classified as Aquitard down perimeter (second column) and the distances between the matched points.

Relational matching: Aquiclude and Aquitard down		
Aquiclude ID	**Aquitard down ID**	**Distance (m)**
1006	1246	2837.64585704418
1007	1246	2839.08418508504
1102	1307	1213.87685124975
1327	1327	69.4
1307	1102	68
1264	1007	44
1246	1006	3566.00126191789
1250	1006	1573.6927908585
1252	1128	1368.80630112518
1076	1076	7315.52088644411
1082	1076	5586.08102787634
1128	1252	4316.76707733924
1352	1252	526.805504906697

Table 7.15 Relational matching: Aquiclude and Aquitard down

Figure 8.34 shows the 12 red points belonging to the Aquiclude perimeter layer, the 13 blue points belonging to the Aquitard down perimeter and the connections between the different points in blue. The numbers in black represent the ID of each point.

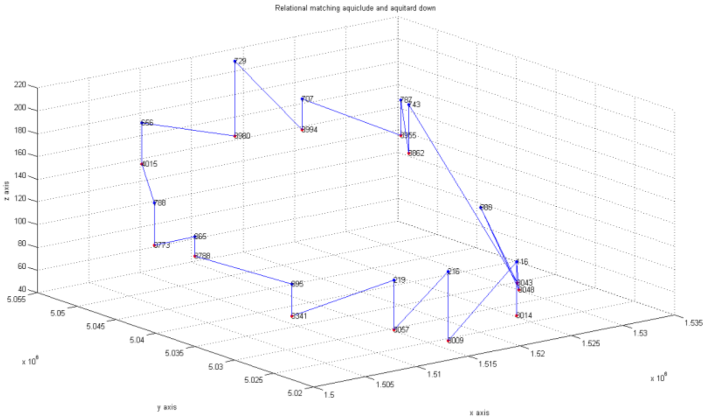


Figure 8.34 Relational matching: Aquiclude and Aquitard down

The fifth step is completed and it is possible to go to the next one.

7.8. INTERPOLATION

Sixthly, it is necessary to obtain new information from the Delaunay triangulation. This procedure is continued using the interpolation method. As explained in chapter 2, section 7, the new points are found using the parametric equation of a line passing through two points. In this specific case, the software generates 5 new points starting from two input points.

In the following, the interpolation of each cluster is described. The software provides four tables and four 3D plots, one for each layer.

7.8.1.Land surface

The input files needed are:

- “LSTriangulation.txt”, and
- “LSPerimeter.txt”.

The output file given is “NewPointsLS.txt”. The 3D output plot given is “New points Land surface.tif”.

Figure 8.35 shows the points belonging to the land surface layer in red, the new points in blue, and the perimeter in blue.

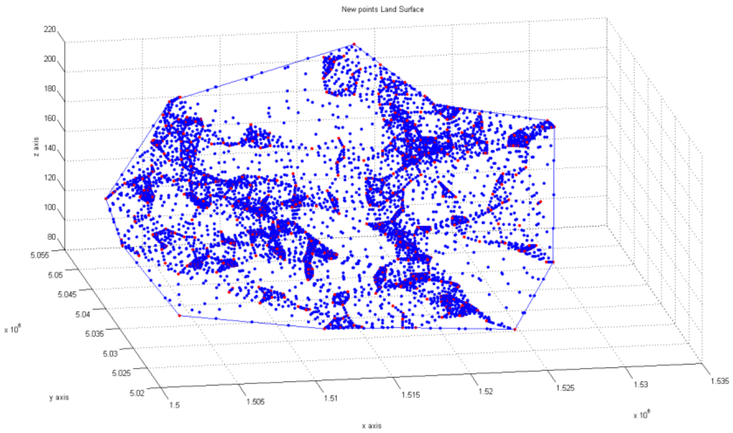


Figure 8.35 Land surface

The number of the new points is 5916.

7.8.2.Aquitard up

The input files needed are:

- “AUTriangulation.txt”, and
- “AUPerimeter.txt”.

The output file given is “NewPointsAU.txt”. The 3D output plot given is “New points Aquitard up.tif”.

Figure 8.36 shows the points belonging to the Aquitard down layer in red, the new points in blue, and the perimeter in blue.

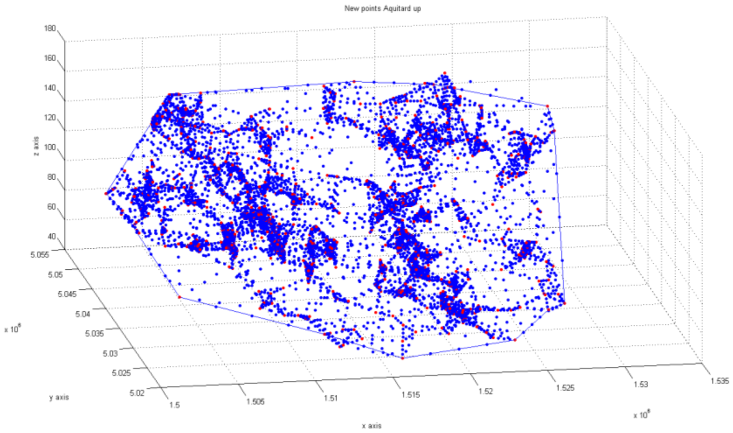


Figure 8.36 Aquitard up

The number of the new points is 5722.

7.8.3.Aquitard down

The input files needed are:

- “ADTriangulation.txt”, and
- “ADPerimeter.txt”.

The output file given is “NewPointsAD.txt”. The 3D output plot given is “New points Aquitard down.tif”.

Figure 8.37 shows the points belonging to the Aquitard down layer in red, the new points in blue, and the perimeter in blue.

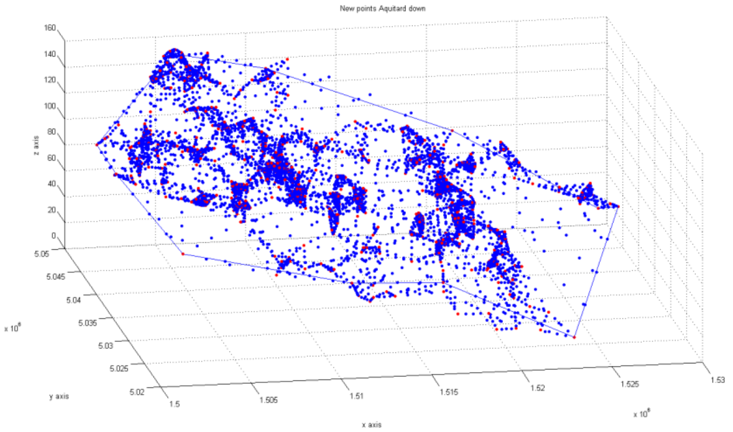


Figure 8.37 Aquitard down

The number of the new points is 4308.

7.8.4.Aquiclude

The input files needed are:

- “AQTriangulation.txt”, and
- “AQPerimeter.txt”.

The output file given is “NewPointsAQ.txt”. The 3D output plot given is “New points Aquiclude.tif”.
Figure 8.38 shows the points belonging to the Aquiclude layer in red, the new points in blue, and the perimeter in blue.

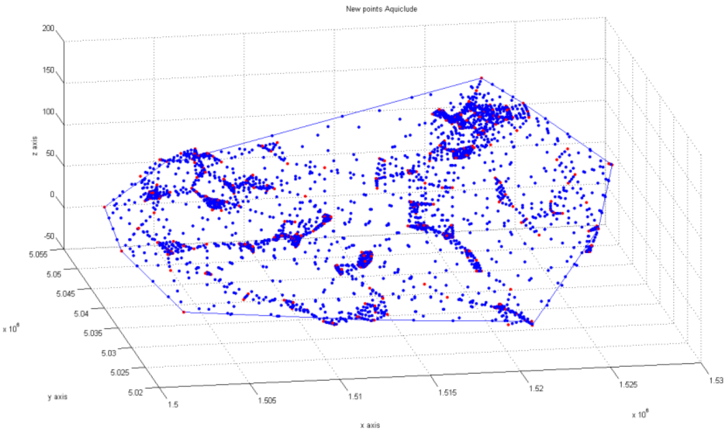


Figure 8.38 Aquiclude

The number of the new points is 2092.

7.9. NEW POINTS FROM THE POINT CLOUD

Finally, it is possible to obtain new information from the data. Starting from the planimetric coordinates given in input, the software generates the height of the point. This procedure uses the average. In fact, the height of the needed point is found using the average values of the height of the points closely located.
In the following, four examples of new points are described. They are one for each cluster.

7.9.1.Land surface

The input files needed are:

- “LSCluster.txt”,
- “LSPerimeter.txt”, and
- “NewPointsLS.txt”.

The input coordinates are the ones provided by the user. The output information is the height of the point and the 3D output plot is a *.tif file.
The input coordinates chosen are:

- North: 1517365 m and
- East: 5031638 m.

The height given is 110.599 m, if 10 points are used. The height given is 110.436 m, if 5 points are used. The height given is 110.4995 m, if 20 points are used. The two following pictures show the result using 10 points.

Figure 8.39 shows the points belonging to the land surface cluster in red, the perimeter in blue, and the new point in green.

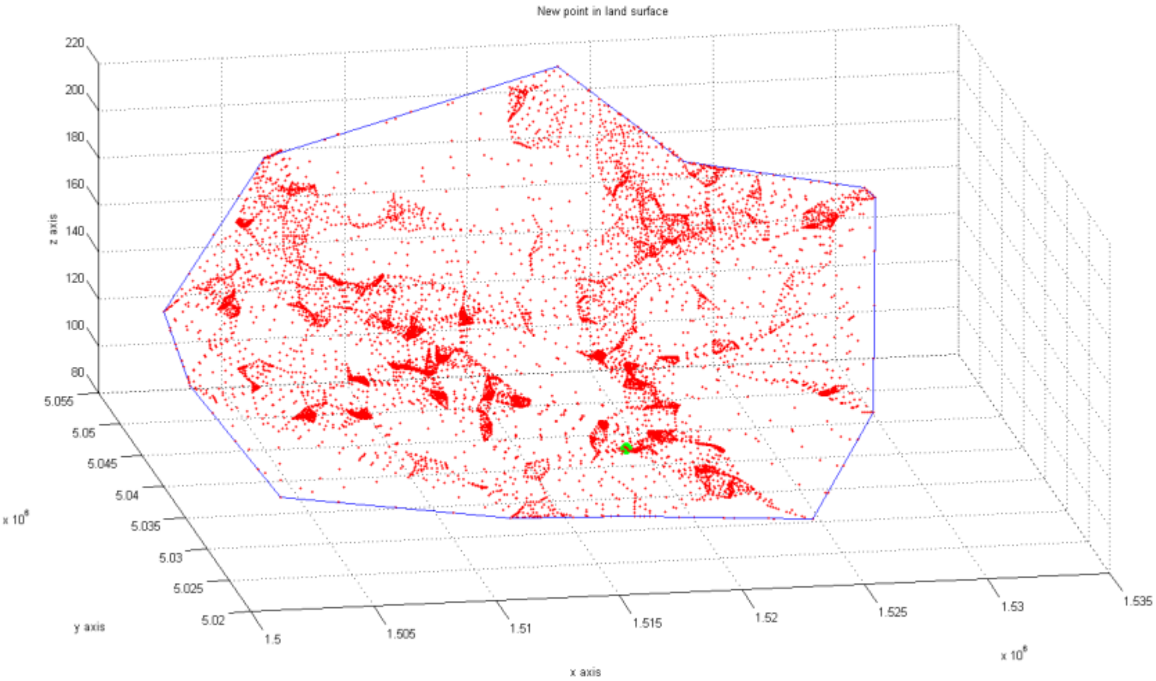


Figure 8.39 Example of a new point in the land surface cluster

Figure 8.40 shows the same 3D plot, from a different point of view. The point is located at the correct height.

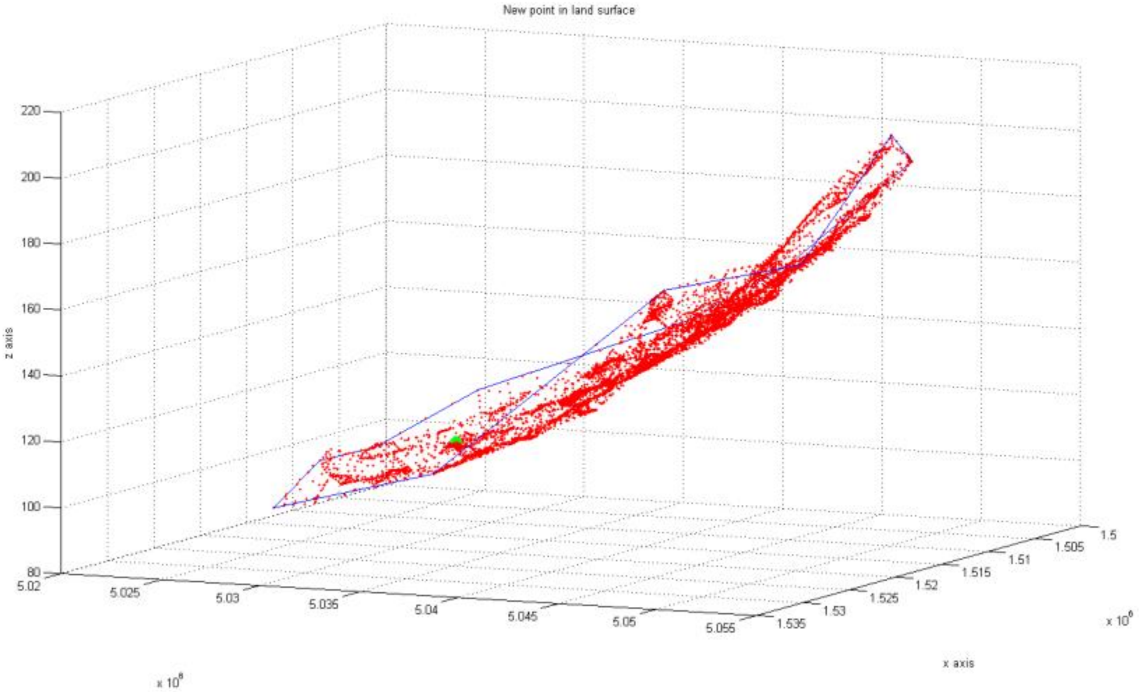


Figure 8.40 Previous example of a new point in the land surface cluster

7.9.2.Aquitard up

The input files needed are:

- “AUCluster.txt”,
- “AUPerimeter.txt”, and
- “NewPointsAU.txt”.

The input coordinates are the ones provided by the user. The output information is the height of the point and the 3D output plot is a *.tif file.

The input coordinates chosen are:

- North: 1517365 m and
- East: 5031638 m.

The height given is 63.776 m, if 10 points are used. The height given is 63.884 m, if 5 points are used. The height given is 57.586 m, if 20 points are used. The two following pictures show the result using 10 points.

Figure 8.41 shows the points belonging to the Aquitard up cluster in red, the perimeter in blue, and the new point in green.

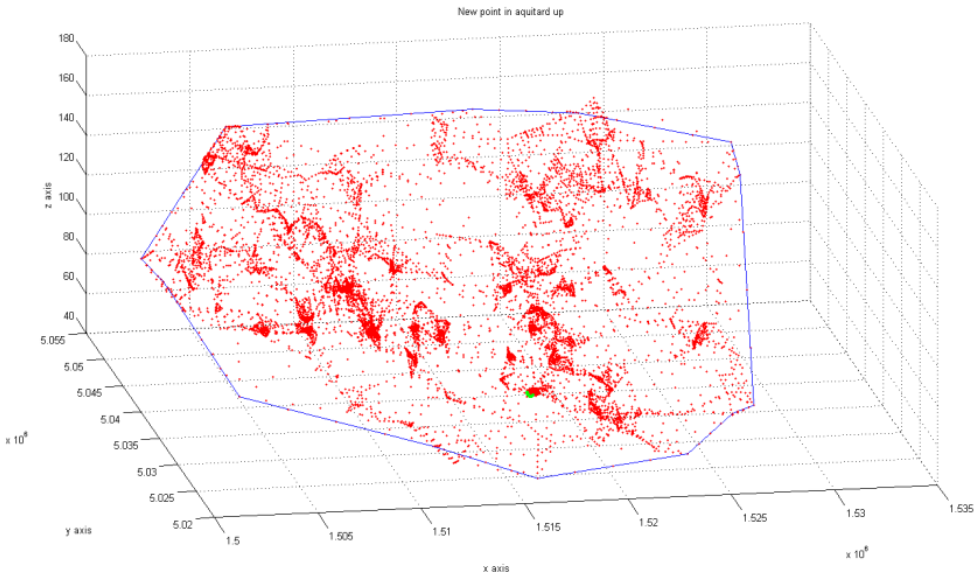


Figure 8.41 Example of a new point in the Aquitard up cluster

Figure 8.42 shows the same 3D plot, from a different point of view. The point is located at the correct height.

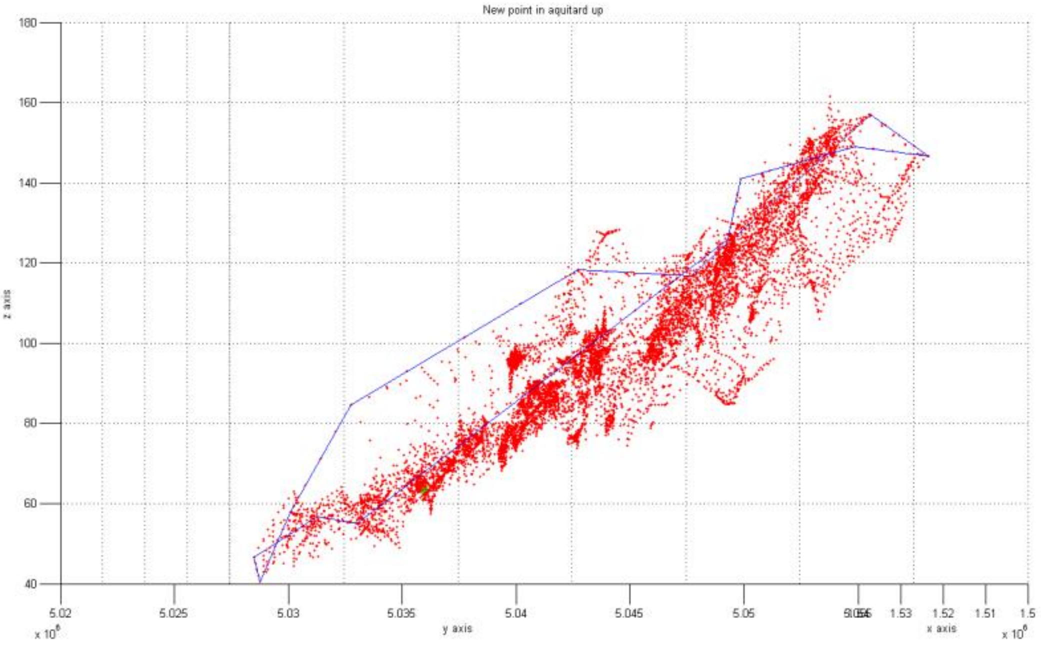


Figure 8.42 Previous example of the new point in the Aquitard up cluster

7.9.3.Aquitard down

The input files needed are:

- “ADCluster.txt”,
- “ADPerimeter.txt”, and
- “NewPointsAD.txt”.

The input coordinates are the ones provided by the user. The output information is the height of the point and the 3D output plot is a *.tif file.
The input coordinates chosen are:

- North: 1517365 m and
- East: 5031638 m.

The height given is 52.724 m, if 10 points are used. The height given is 53.536 m, if 5 points are used. The height given is 53.6745 m, if 20 points are used. The two following pictures show the result using 10 points.
Figure 8.43 shows the points belonging to the Aquitard down layer in red, the perimeter in blue, and the new point in green.

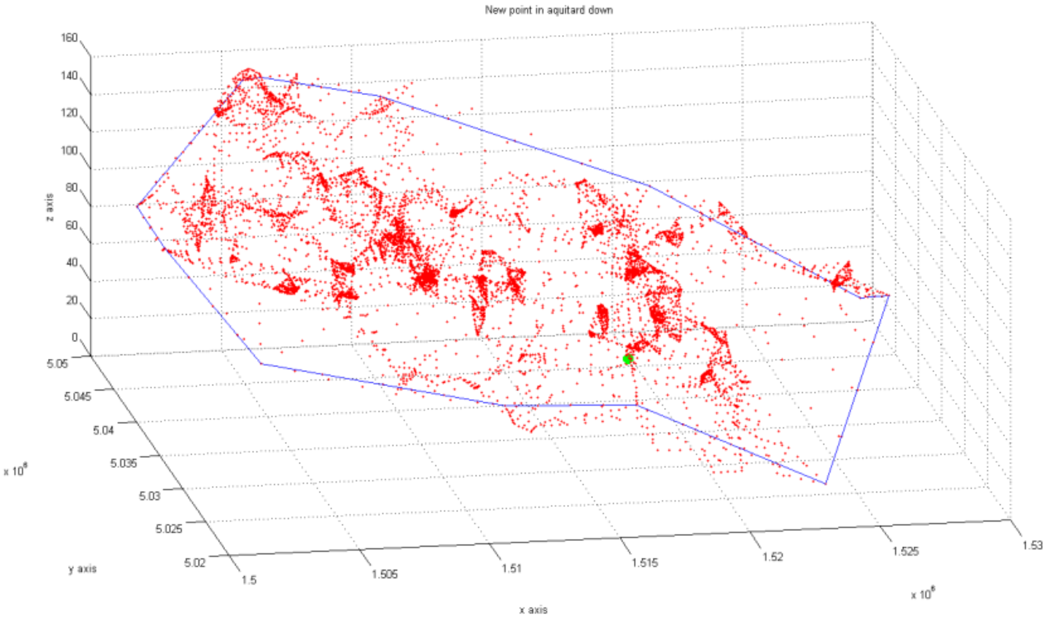


Figure 8.43 Example of a new point in the Aquitard down cluster

Figure 8.44 shows the same 3D plot, from a different point of view. The point is located at the correct height.

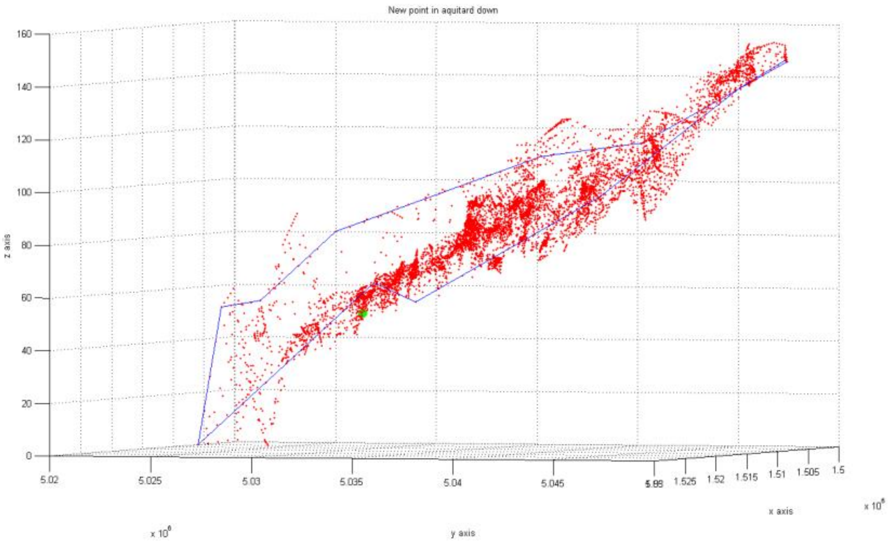


Figure 8.44 Previous example of the new point in the Aquitard down cluster

7.9.4.Aquiclude

The input files needed are:

- “AQCluster.txt”,
- “AQPerimeter.txt”, and
- “NewPointsAQ.txt”.

The input coordinates are the ones provided by the user. The output information is the height of the point and the 3D output plot is a *.tif file.

The input coordinates chosen are:

- North: 1517365 m and
- East: 5031638 m.

The height given is -13.954 m, if 10 points are used. The height given is -13.372001 m, if 5 points are used. The height given is -13.1595 m, if 20 points are used. The two following pictures show the result using 10 points.

Figure 8.45 shows the points belonging to the Aquitard down layer in red, the perimeter in blue, and the new point in green.

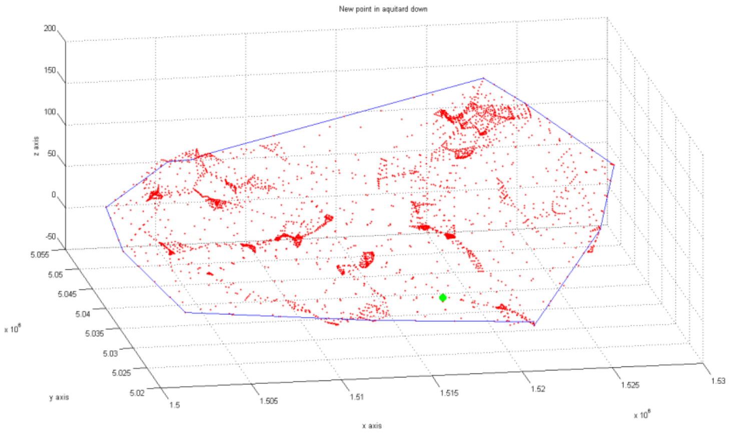


Figure 8.45 Example of a new point in the Aquitard down cluster

Figure 8.46 shows the same 3D plot, from a different point of view. The point is located at the correct height.

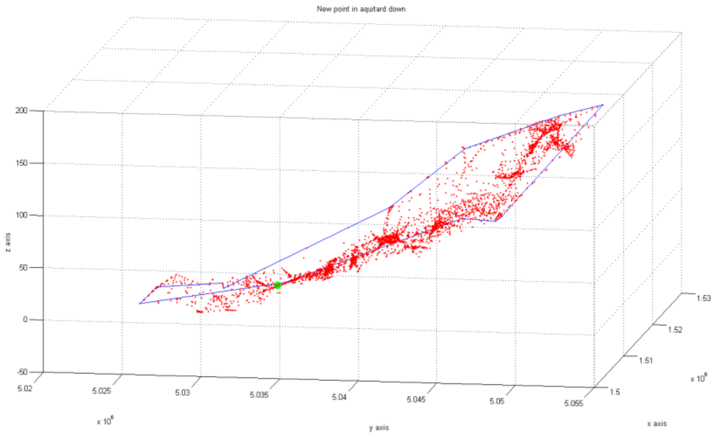


Figure 8.46 Previous example of the new point in the Aquitard down cluster

CHAPTER 9

The fourth case study:
the shrub. Levy county and Marion county, in Florida, U.S.A.

Staring from a LiDAR point cloud, the purposes of this chapter are two: the former is to obtain a better classification of the input data and the latter is to obtain density maps.

All the input points are classified into three classes: class 1 includes unclassified points, comprising vegetation, buildings, noise, etc; class 2 includes ground points; and class 9 includes water points. It is possible to better classify class 1 into new sets: roads and vegetation. This last set contains a sub-set called shrub. A shrub or bush is a woody plant characterized by several stems arising from the base and height usually under 19.658 ft (6 m) tall.

Once the shrub is detected, the idea is to cluster the shrub into subsets according to the location and the density. The shrub can be divided into three type of density: light, medium and heavy. Density maps can be used by rescuers during rescue emergency. It is possible to search for lost people using also rescuers and scenting dog. It has been documented that one well trained dog team can be effective as 50 ground searchers.

This work can be used as the input of fire disaster pre-plan. Indeed mathematical fire behavior models require information on the location, the density and the species of each tree. Using LiDAR data, it is possible to detect the location, the height and the density, but not the species.

The following subchapters explain in details the method adopted.

9.1. LEVY COUNTY

The first study area is located in Levy County, Florida, U.S.A. According to the 2000 census, the county has a total area of 1,412.32 square miles (3,657.9 km^2), of which 1,118.38 square miles (2,896.6 km^2) (or 79.19%) is land and 293.94 square miles (761.3 km^2) (or 20.81%) is water [Wikipedia].

In figure 9.1 is displayed Levy County location.

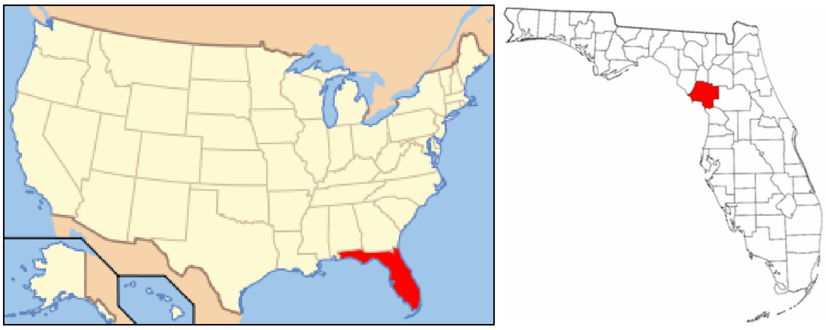


Figure 9.1 Levy County, Florida, USA

The input data comprise three sets:

1. raw point cloud LiDAR data,
2. breaklines, and
3. metadata.

The input point cloud is a raw point cloud data in LAS 1.1 format. LiDAR points were acquired between July 1, 2007 and August 16, 2007. LiDAR point cloud meets 3.28 ft (1 m) horizontal accuracy standard at the 95% confidence level. LiDAR is required to meet 0.60 ft Fundamental Vertical Accuracy (FVA) and 1.19 ft Consolidated Vertical Accuracy (CVA). Countywide LiDAR dataset tested 0.58 ft vertical accuracy at 95% confidence level in open terrain, based on open terrain RMSEz (0.30 feet) * 1.9600. Countywide LiDAR dataset tested 0.85 ft vertical accuracy at 95% confidence level in all land cover categories combined, based on consolidated RMSEz (0.435 feet) * 1.9600.
The raw point cloud LiDAR is divided into 733 tiles. Each one is identify by the tile_ID.
The coordinates NAD 1983 HARN StatePlane Florida West FIPS 0902 Feet, the projection is the Trasverse Mercator, the Datum is the North American 1983 HARN. The unit system is US foot.

The input data contain also breaklines. They are coastal shoreline, contour_1ft, contour_2ft, footprint, ground control, hydrographic feature, single line feature1, dual line feature, island, low confidence, mass point, overpass, road breaklines, soft feature, vertacctestpts, water body. A better description of each breaklines is provided in the following list:

- COASTALSHORELINE: the coastal breakline delineates the land water interface using LiDAR data as reference.
- CONTOUR_1ft: contours consistent with the deliverables and including a certain level of smoothing. Index, intermediate, depression, supplementary and low confidence contours are coded.
- CONTOUR_2ft: contours consistent with the deliverables and including a certain level of smoothing. Index, Intermediate, Depression, Supplementary and Low confidence contours are coded.
- FOOTPRINT: it defines the deliverable area using the official Florida tiling scheme.

- GROUNDCONTROL: the purpose of this Geodetic Control Survey was to provide accurate and consistent horizontal and vertical network control for LiDAR and Photogrammetric mapping using Global Positioning System (GPS) technology.
- HYDROGRAPHICFEATURE:
    - Single Line Feature1: linear hydrographic features such as streams, shorelines, canals, swales, embankments, etc. with an average width less than or equal to 8 feet. In the case of embankments, if the feature forms a natural dual line channel, then capture it consistent with the capture rules. Other embankments fall into the soft breakline feature class.
    - Dual Line Feature: linear hydrographic features such as streams, shorelines, canals, swales, etc. with an average width greater than 8 feet. In the case of embankments, if the feature forms a natural dual line channel, then capture it consistent with the capture rules. Other embankments fall into the soft breakline feature class. These hydrographic features are hydro-enforced and maintain monotonicity. Where dry drainage features were collected with a hydrographic feature, these breaklines are not hydro-enforced but deliberately include undulations that more accurately represent true topography. Dry drainage features collected in the hydrographic feature class have a sub-class code of soft hydro. This enables hydro-enforced hydrographic features to be distinguished from non-hydro-enforced soft hydrographic features representing dry drainage features.
- ISLAND: apparent boundary of natural or man-made island feature captured with a constant elevation. Island features will be captured for features one-half acres in size or greater.
- LOWCONFIDENCE: apparent boundary of vegetated areas that are considered obscured to the extent that adequate vertical data cannot be clearly determined to accurately define the DTM. These features are for reference only to indicate areas where the vertical data may not meet the data accuracy requirements due to heavy vegetation.
- MASSPOINT: LiDAR generated ground points.
- OVERPASS: bridge or overpass features. Feature should show edge of bridge or overpass.
- ROADBREAKLINE: road feature. It capture edge of pavement (non-paved or compact surfaces as open to compiler interpretability) on both sides of the road. Runways are not to be included.
- SOFTFEATURE: supplemental breaklines where LiDAR mass points are not sufficient to create a hydrologically correct DTM. Soft features shall include ridges, valleys, top of banks, etc. Soft features may also include natural embankments that act as small ponding areas. Top of Banks can also be included in the soft breakline class so long as it does not define the edge of a water feature.
- VERTACCTESTPTS: these checkpoints were acquired by an indenpendent survey in order to produce a vertical accuracy report of the LiDAR data. The report will be based on a minimum of 30 ground measurements for each of four land cover categories, totaling 120 test points for each 500 square mile area of new topographic data collection. The land cover measurements distributed through each project area will be collected for each of the following land cover categories:
    - bare-earth and low grass;
    - brush Lands and low trees;

  - forested areas fully covered by trees;
  - urban areas.
- WATERBODY: they are closed water bodies: land/water boundaries of constant elevation water bodies such as lakes, reservoirs, ponds, etc. Features shall be defined as closed polygons and contain an elevation value that reflects the best estimate of the water elevation at the time of data capture. Water body features are be captured for features one-half acres in size or greater. Donuts will exist where there are islands within a closed water body feature.

The following image shows the input breaklines provided for the entire study area.

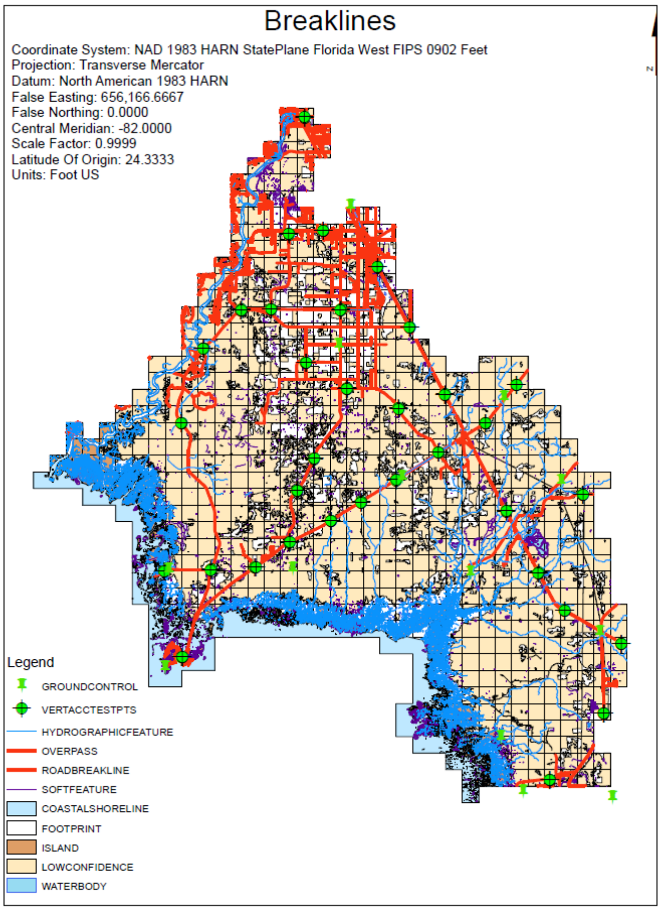

Breaklines
Coordinate System: NAD 1983 HARN StatePlane Florida West FIPS 0902 Feet
Projection: Transverse Mercator
Datum: North American 1983 HARN
False Easting: 656,166.6667
False Northing: 0.0000
Central Meridian: -82.0000
Scale Factor: 0.9999
Latitude Of Origin: 24.3333
Units: Foot US
N
Legend
GROUNDCONTROL
VERTACCTESTPTS
HYDROGRAPHICFEATURE
OVERPASS
ROADBREAKLINE
SOFTFEATURE
COASTALSHORELINE
FOOTPRINT
ISLAND
LOWCONFIDENCE
WATERBODY

Figure 9.2 Breaklines

All the results will be presented for the tile #35712. The following picture shows the breaklines for this particular tile. The footprint is in turquoise, the hydrographic feature in light blue, low confidence zones in beige, the overpass in red and the breaklines in red too.

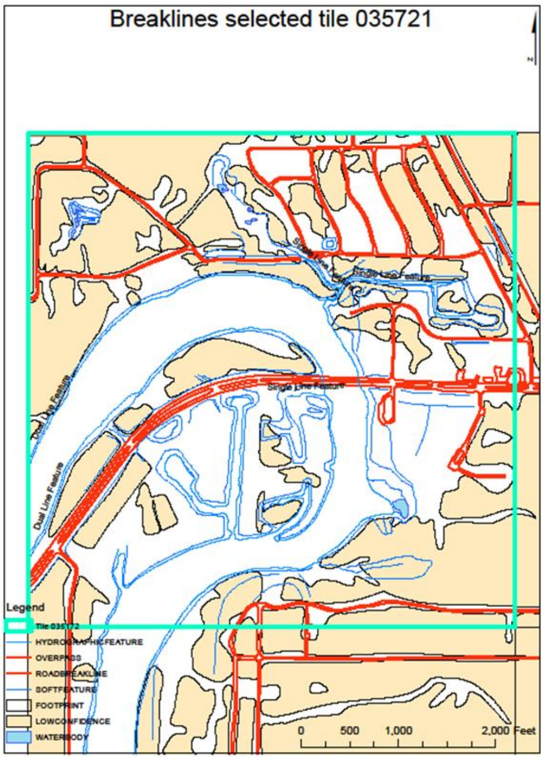


Figure 9.3 Breaklines of tile #35712

The input data contain also the metadata. They are divided into 15 file (14 breaklines and LAS) and they are:

- FGDC_FDEM_Levy_COASTALSHORELINE
- FGDC_FDEM_Levy_CONTOUR_1FT
- FGDC_FDEM_Levy_CONTOUR_2FT
- FGDC_FDEM_Levy_FOOTPRINT
- FGDC_FDEM_Levy_GROUNDCONTROL
- FGDC_FDEM_Levy_HYDROGRAPHICFEATURE
- FGDC_FDEM_Levy_ISLAND
- FGDC_FDEM_Levy_LAS
- FGDC_FDEM_Levy_LOWCONFIDENCE
- FGDC_FDEM_Levy_MASSPOINT
- FGDC_FDEM_Levy_OVERPASS
- FGDC_FDEM_Levy_ROADBREAKLINE
- FGDC_FDEM_Levy_SOFTFEATURE
- FGDC_FDEM_Levy_VERTACCTESTPTS
- FGDC_FDEM_Levy_WATERBODY

9.2. METHODOLOGY

In order to sub-classify class 1, different tools in this specific order have to be used:

1. LAS to multipoints: this tool imports all the files in LAS format into a new multipoint feature class. File formats supported are LAS 1.0, LAS 1.1 and LAS 1.2. The input data is in

LAS 1.1 format. The LAS to multipoints step is applied twice: the first time to obtain the non-ground points using the Model Builder "LAStoMPng". The second time to obtain the ground points using the Model Builder "LAStoMPg".

2. Point to raster: the point to raster converts point features to a raster dataset. The point to raster step is applied twice: the first time to obtain the Digital Surface Model (DSM), the second time to obtain the Digital Elevation Model (DEM). The Point to raster step is applied using the Model Builder "NGtoDSM" and "GtoDEM".
3. This step is used to obtain the relative height of all the points. The a python script "nDSM" is used.
4. Raster to point: the raster to point conversion tool converts a raster dataset to point features. The Raster to Point Conversion tool converts a raster dataset to point features. For each cell of the input raster dataset, a point will be created in the output feature class. The points will be positioned at the centers of cells that they represent. The NoData cells will not be transformed into points. To convert the raster into points, the python "RasterToPoint" script is used.
5. Select by attributes: it adds, updates, or removes a selection on a layer or table view based on an attribute query. To select all the points within a certain height, Select Layer By Attribute (Data Management) is used. This step is applied four times:
    a. vegetation high lower than 0 ft
    b. Height between 0 ft and 3.28 ft
    c. Height between 3.28 ft and 6.56 ft
    d. Height between 6.56 ft and 19.658 ft

    These steps are carried on using the "LowerThanZero", "LowerThanThree", "LowerThanSix" and "LowerThanNinteen" python scripts.
6. Split: splitting the input features, it creates a subset of multiple output feature classes.
7. Feature to polygon: it creates a feature class containing polygons generated from areas enclosed by input line or polygon features.
8. Clip: it extracts input features that overlay the clip features. This tool cuts out a piece of the roads feature class using the footprint feature in the new feature class called clipped roads.
9. Selects features in a layer based on a spatial relationship to features in another layer.
10. Point density: it calculates a magnitude per unit area from point features that fall within a neighborhood around each cell.

9.3. RESULTS AND MAPS: Levy County

The total number of the tiles is 733. All the results and the maps will be displayed for tile # 35712.

9.3.1. LAS to multipoints

9.3.1.1. Non-ground

The parameters setting list is provided:

The input is the LAS folder.
The output feature class is the non-ground folder

The average point spacing is 4.
The class code is 1 and 2.
The return value is FIRST RETURN.
The input coordinate system is NAD 1983 HARN State Plane Florida West FIPS 0902 Feet.
The file suffix is las.
The z factor is 1.

9.3.1.2. Ground

The parameters setting list is provided:

The input is the LAS folder.
The output feature class is the ground folder.
The average point spacing is 4.
The class code is 2.
The return value is ANY RETURN.
The input coordinate system is NAD 1983 HARN State Plane Florida West FIPS 0902 Feet.
The file suffix is las.
The z factor is 1.

9.3.2. Point to raster

9.3.2.1. DSM

The parameters setting list is provided:

The input feature is the non-ground folder.
The value field is Z.
The output feature is the DSM folder.
The cell assignment is the maximum value of the attributes of the points within the cell.
The priority field is none.
The cell size is 5 ft.
The following image shows the DSM, tile #35712.

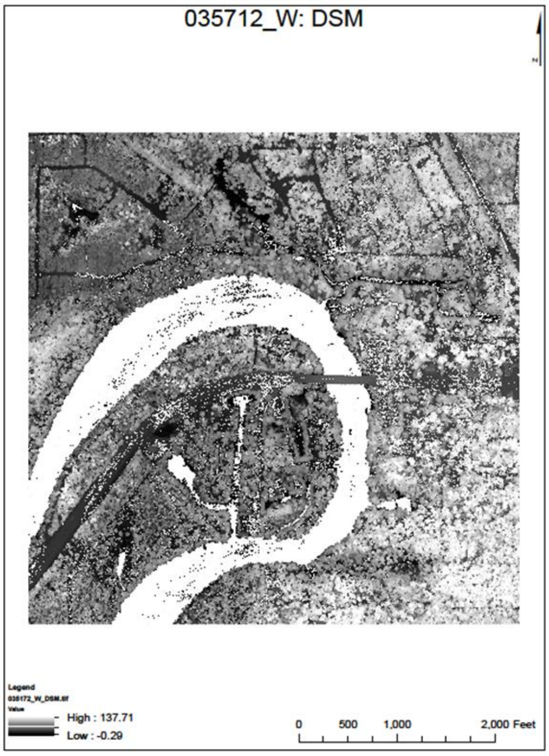


Figure 9.4 DSM, tile #35712

9.3.2.2. DEM

The parameters setting list is provided:

The input feature is the ground folder.
The value field is Z.
The output feature is the DEM folder.
The cell assignment is the maximum value of the attributes of the points within the cell.
The priority field is none.
The cell size is 5 ft.
The following image shows the DEM, tile #35712.

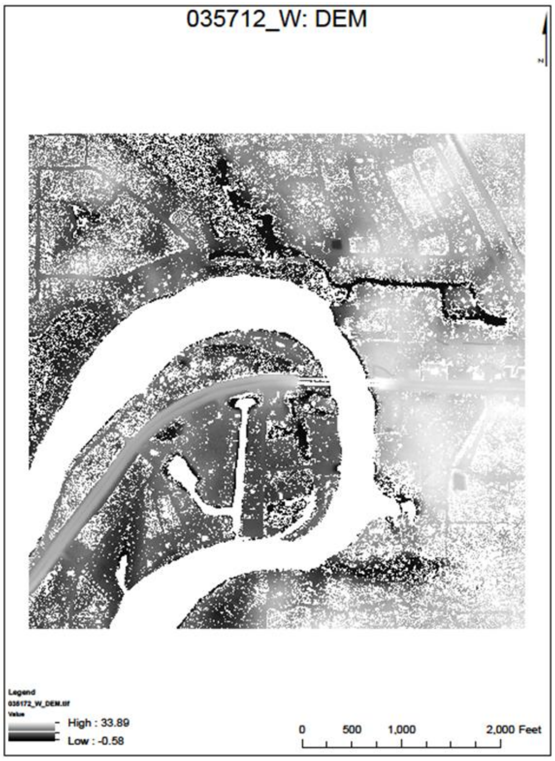


Figure 9.5 DEM, tile #35712

9.3.3. nDSM: DSM - DEM

The parameters setting list is provided:

The following image shows the nDSM, tile #35712.

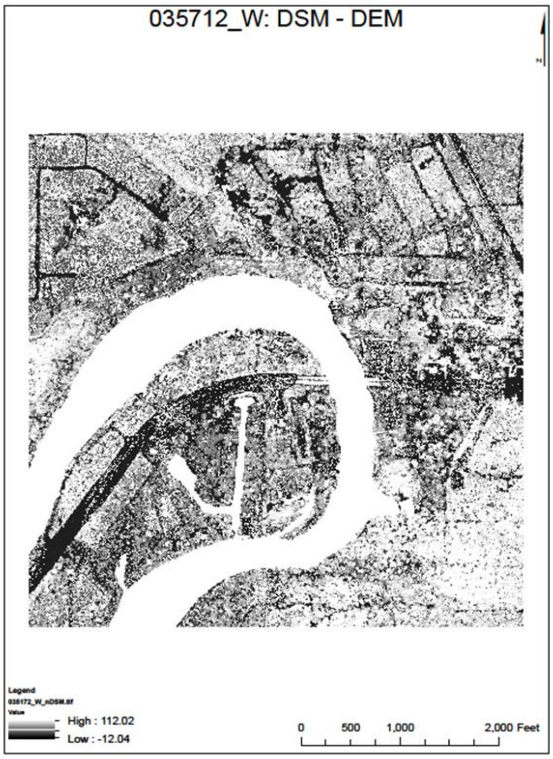


Figure 9.6 DSM - DEM, tile #35712

9.3.4. Raster to point

The parameters setting list is provided:

The input raster is the nDSM folder.
The output point features is the OutputPoints folder.

9.3.5. Select layer by attribute

9.3.5.1. Points lower than 0 ft

The parameters setting list is provided:

The input folder is the OutputPoints.
The output folder is the LowerThan0.
The selection type is the "NEW_SELECTION".
The where clause is the following: "grid_code <= 0 ft".

The following image shows the points lower than 0 ft, tile #35712.

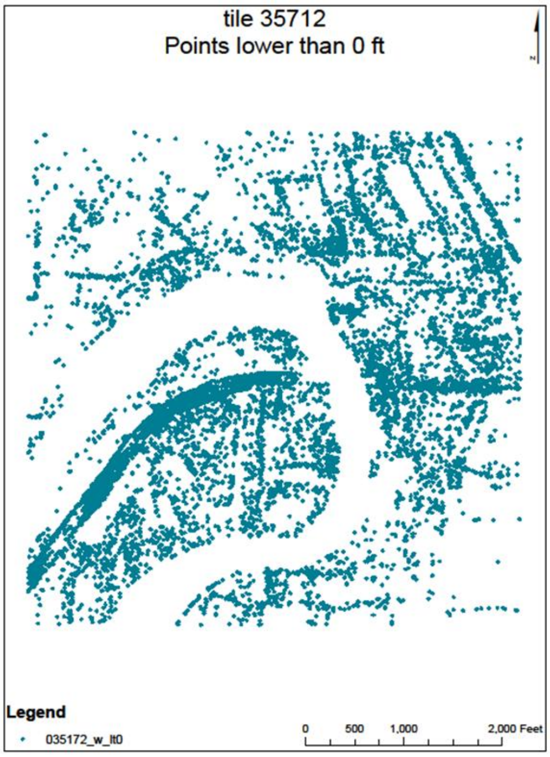


Figure 9.7 Points lower than 0 ft, tile #35712

9.3.5.2. Points between 0 ft and 3.28 ft (1 m)

The parameters setting list is provided:

The input folder is the LowerThan0.
The output folder is the LowerThan3.
The selection type is the "NEW_SELECTION".

The where clause is "grid_code >= 3.28 ft".

The following image shows the points between 0 ft and 3.28 ft, tile #35712.

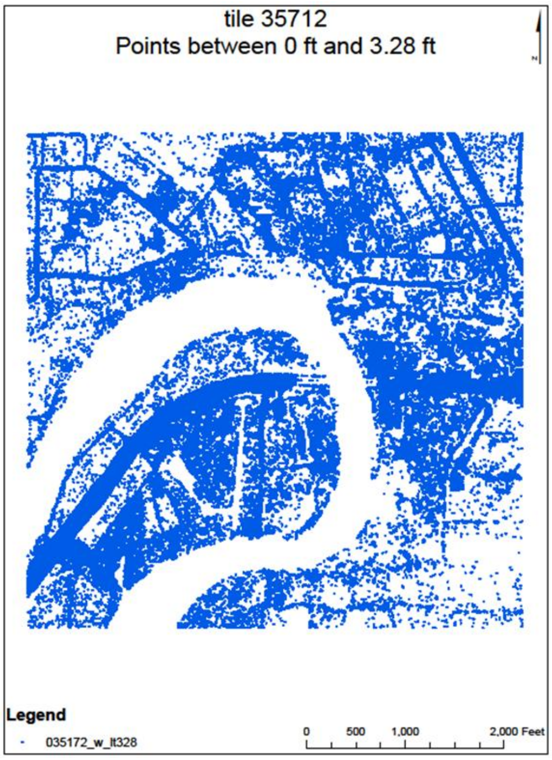


Figure 9.8 Points between 0 ft and 3.28 ft, tile #35712

9.3.5.3. Points between 3.28 ft (1 m) and 6.56 ft (2 m)

The parameters setting list is provided:

The input folder is the LowerThan3.
The output folder is the LowerThan6.
The selection type is the "NEW_SELECTION".
The where clause is "grid_code >= 6.56 ft".

The following image shows the points between 3.28 ft and 6.56 ft, tile #35712.

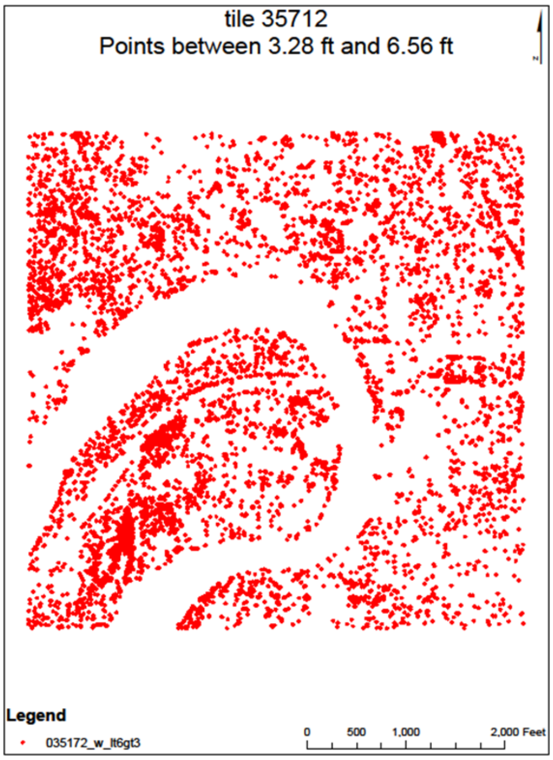


Figure 9.9 Points between 3.28 ft and 6.56 ft, tile #35712

9.3.5.4. Points between 6.56 ft (2 m) and 19.568 ft (6 m)

The parameters setting list is provided:

The input folder is the LowerThan6.
The output folder is the LowerThan19.
The selection type is the "NEW_SELECTION".
The where clause is "grid_code >= 19.658 ft".

The following image shows the points between 6.56 ft and 19.658 ft, tile #35712.

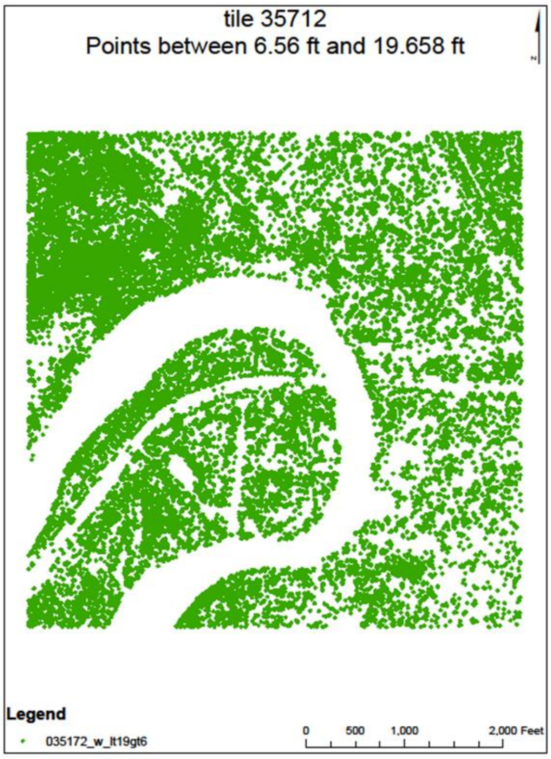


Figure 9.10 Points between 6.56 ft and 19.658 ft, tile #35712

9.3.6. Split

Splitting the Input Features creates a subset of multiple output feature classes.

9.3.6.1. Roads

The parameters setting list is provided:

The input feature is the road breaklines.
The split feature is the footprint breaklines.
The split field is the cell number.
The cluster tolerance is 0 ft.
The output feature is the split roads.

9.3.6.2. Hydrograph

The parameters setting list is provided:

The input feature is the hydrograph breaklines.
The split feature is the footprint breaklines.
The split field is the cell number.
The cluster tolerance is 0 ft.
The output feature is the split hydrograph.

9.3.7. Feature to polygon

The parameters setting list is provided:

The input feature is the split roads folder.
The output feature class is the ftp_road folder.
The cluster tolerance is 0.2 ft.
There are "NO ATTRIBUTES".

The following image shows the roads in tile #35712.

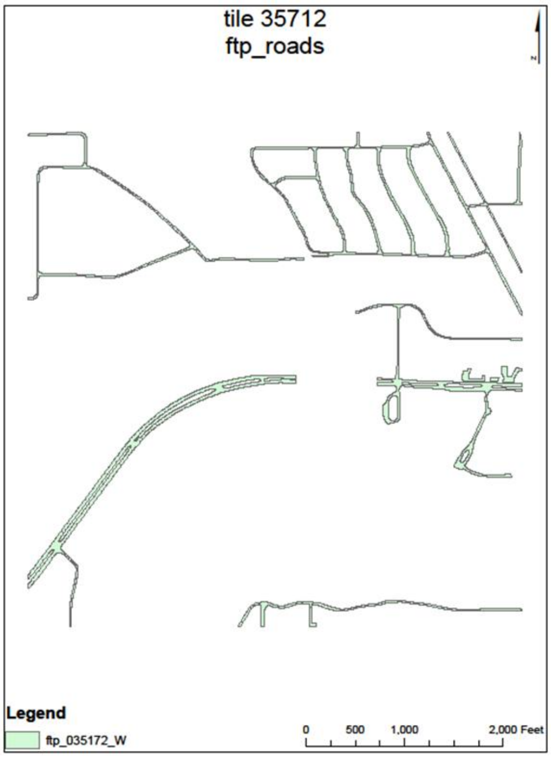


Figure 9.11 Points between 6.56 ft and 19.658 ft, tile #35712

9.3.8. Clip

9.3.8.1. Roads between 0 ft and 3.28 ft

The parameters setting list is provided:

The input features is the LowerThan3 folder.
The clip features is the split roads.
The output feature class is Roadpoints
The cluster tolerance is 0 ft.

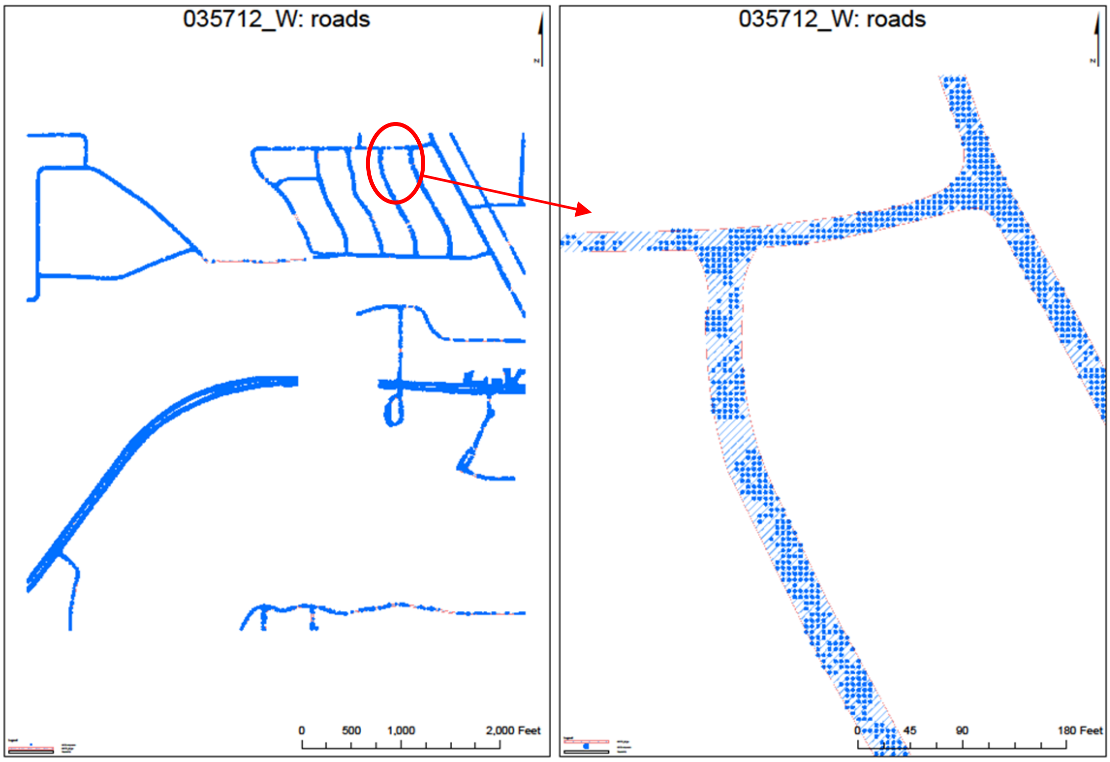


Figure 9.12 (a) Points classified as roads 0 - 3.28 ft, tile #35712
(b) Zoom in: Points classified as roads 0 - 3.28 ft, tile #35712

9.3.8.2. Roads between 3.28 ft and 6.56 ft

The parameters setting list is provided:

The input features is the LowerThan6 folder.
The clip features is the split roads.
The output feature class is Roadpoints
The cluster tolerance is 0 ft.

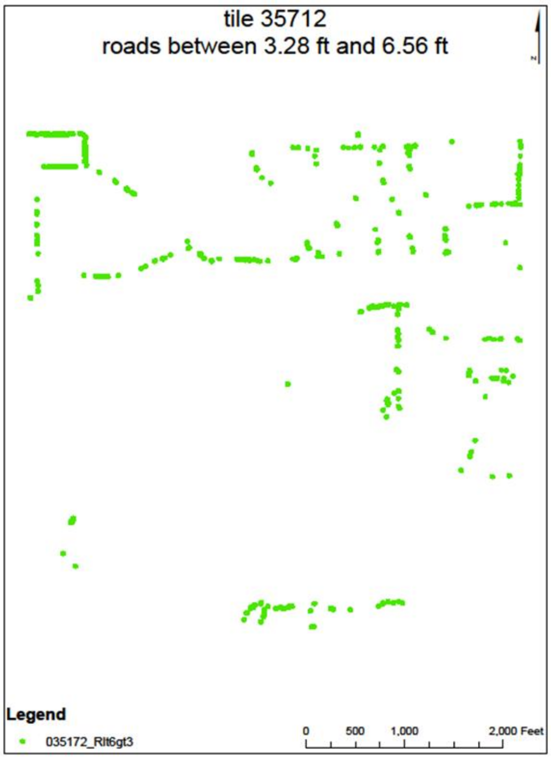


Figure 9.13 Points classified as roads 3.28 - 6.56 ft, tile #35712

9.3.8.3. Roads between 6.56 ft and 19.658 ft

The parameters setting list is provided:

The input features is the LowerThan19 folder.
The clip features is the split roads.
The output feature class is Roadpoints
The cluster tolerance is 0 ft.

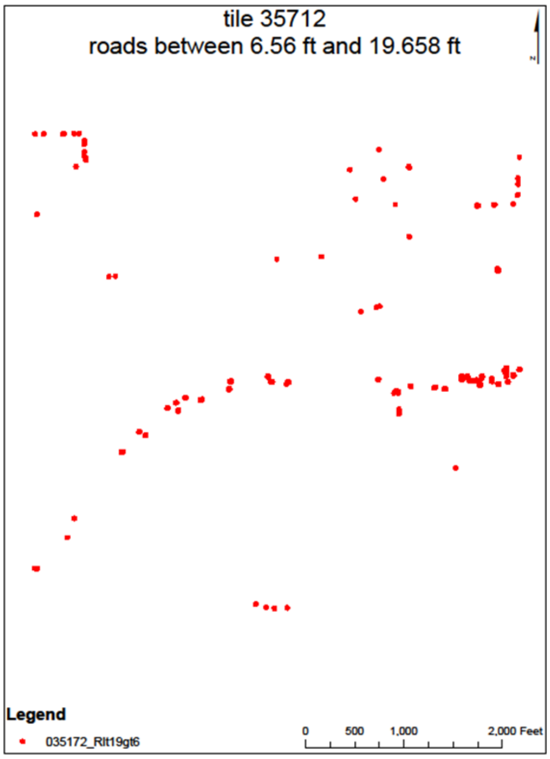


Figure 9.14 Points classified as roads 6.56 - 19.658 ft, tile #35712

9.3.9. Select by Location

Each feature in the Input Feature Layer is evaluated against the features in the Selecting Features layer or feature class; if the specified Relationship is met, the input feature is selected.

9.3.9.1. <u>Vegetation between 0 ft and 3.28 ft</u>

The parameters setting list is provided:

The input layer is PointsLowerThan3.
The source layer is Roadpoints.
The overlap type is “CONTAINS”.
The selection type is "SWITCH_SELECTION".
The output layer is VegetationLowerThan3.

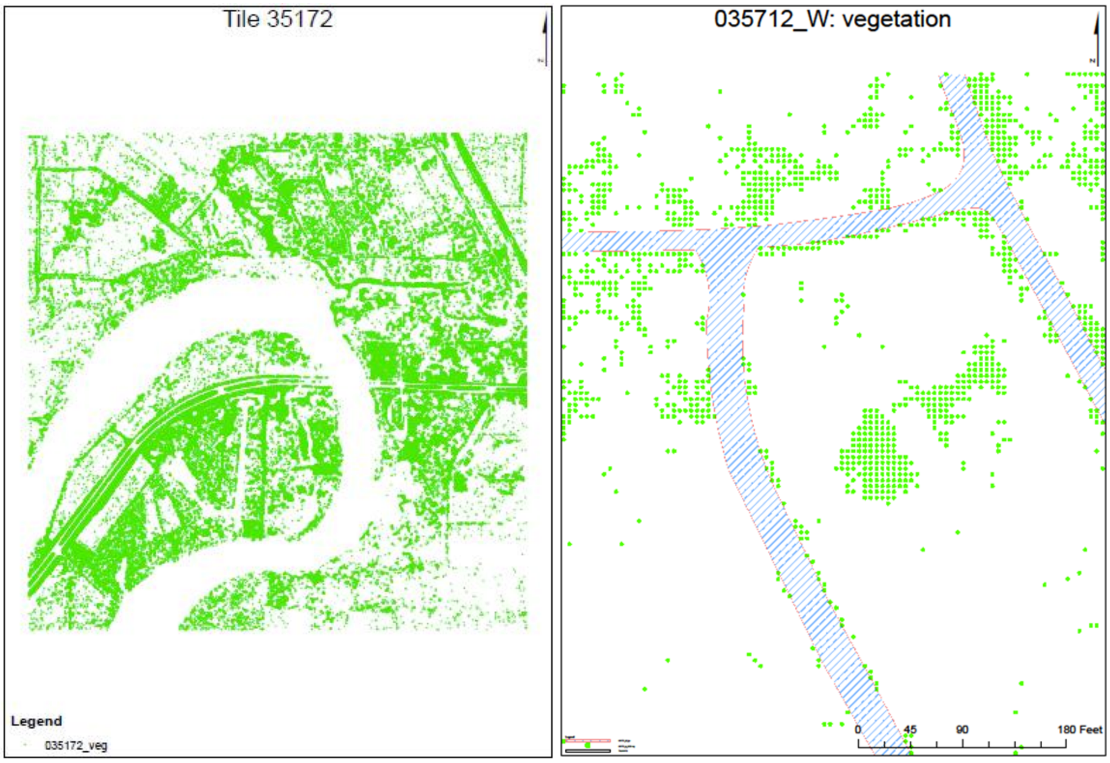


Figure 9.15 (a) Points classified as vegetation, tile #35712
(b) Zoom in: Points classified as vegetation, tile #35712

9.3.9.2. Vegetation between 3.28 ft and 6.56 ft

The parameters setting list is provided:

The input layer is PointsLowerThan6.
The source layer is Roadpoints.
The overlap type is "CONTAINS".
The selection type is "SWITCH_SELECTION".
The output layer is VegetationLowerThan6.

9.3.9.3. Vegetation between 6.56 ft and 19.658 ft

The parameters setting list is provided:

The input layer is PointsLowerThan19.
The source layer is Roadpoints.
The overlap type is "CONTAINS".
The selection type is "SWITCH_SELECTION".
The output layer is VegetationLowerThan19.

9.3.9.4. Vegetation inside wetlands

The parameters setting list is provided:

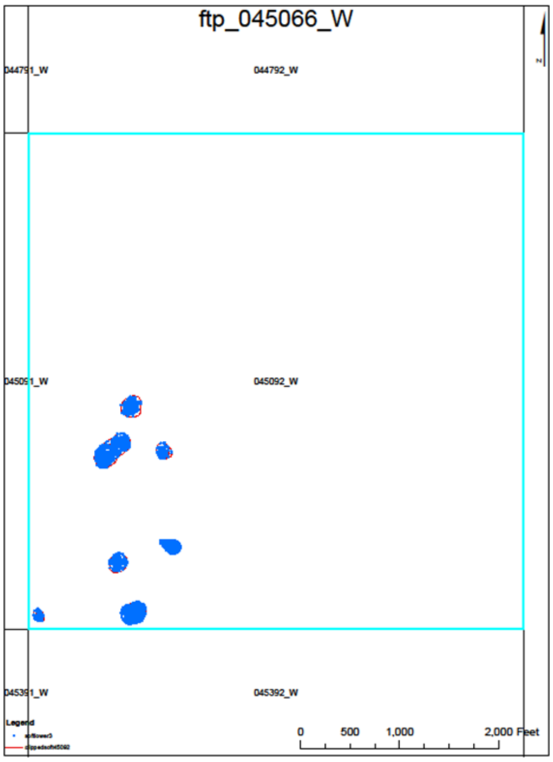


Figure 9.16 Vegetation inside wetland, tile #35712

9.3.10. Point density

The parameters setting list is provided:

The input point features is the VegetationLowerThan3 folder.
The population field is "NONE".
The area unit scale factor is "SQUARE_FEET".
The output raster is the density folder.

Figure X shows the map density, tile #357121. Different colors in the map have different meaning: all the areas without vegetation are in white, all the areas with light density shrub is in green, all the areas with medium density shrub are in yellow, all the areas with high density shrub are in red, all the roads are in gray and the rivers in blue.

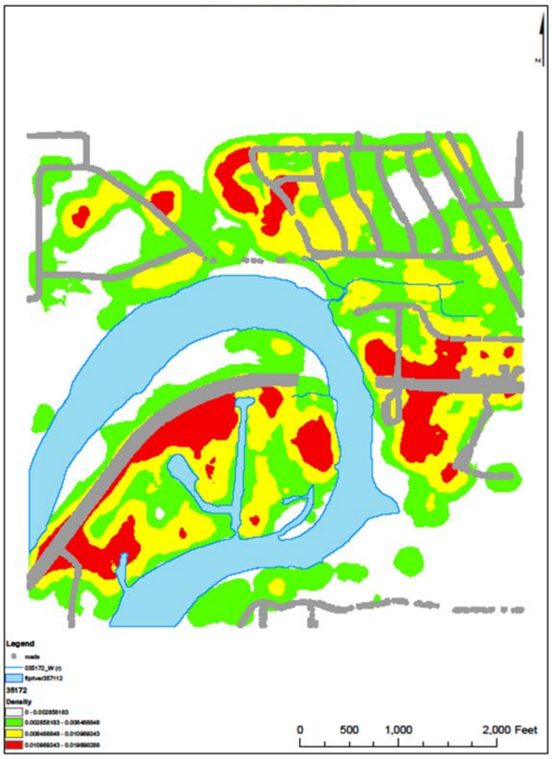


Figure 9.17 Density map of the vegetation, tile #35712

Figure X shows the map density, tile #45066. In red are the soft feature breaklines.

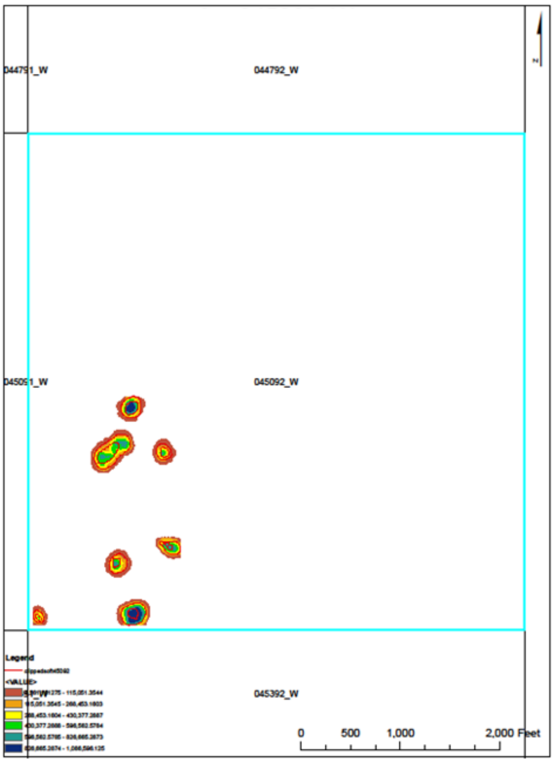


Figure 9.18 Density map of the vegetation, tile #3571

It is possible to obtain a better 3D visualization of the map displayed in figure 9.17 using VNS (see appendix A, section 5).

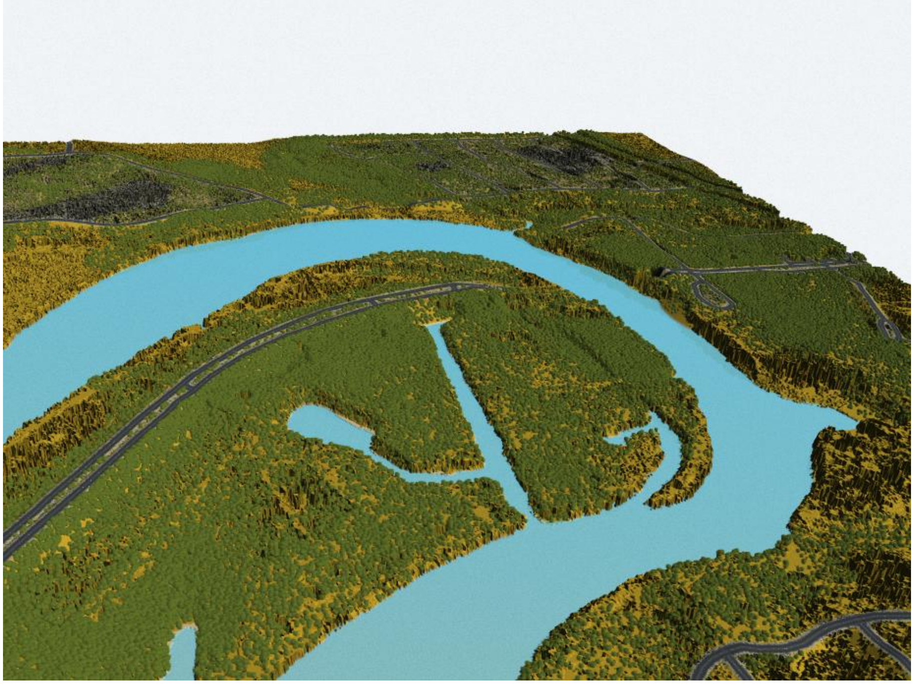


Figure 9.19 Visual Nature Studio, Output #1

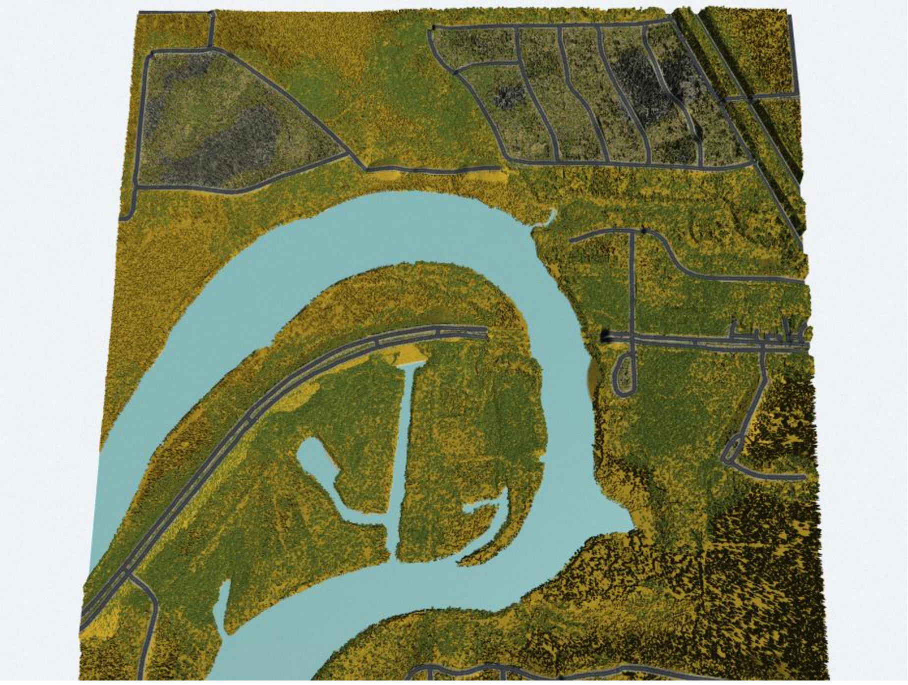


Figure 9.19 Visual Nature Studio, Output #2

9.4. MARION COUNTY AND INPUT DATA

The second study area is located in Marion County, Florida, U.S.A. Census Bureau 2006 estimate for the county is 316,183.

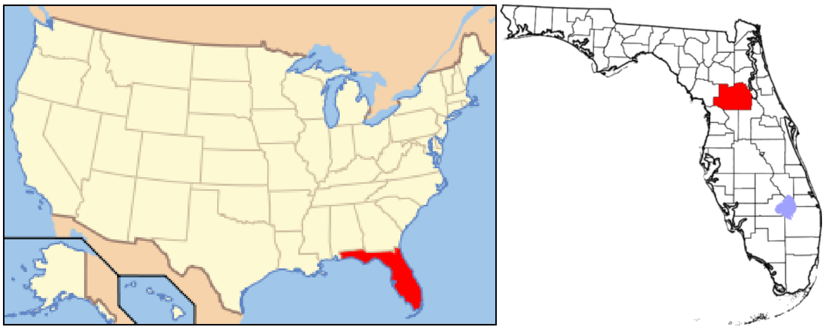


Figure 9.20 Marion County, Florida, USA

As of the census of 2000, there were 258,916 people, 106,755 households, and 74,621 families residing in the county. The population density was 164 people per square mile (63 km^2). There were 122,663 housing units at an average density of 78 per square mile (30 km^2) [Wikipedia].

The area's marshes and shallow seas are among nature's grandest nurseries, hosting seabirds, shore birds, ocean life and numerous estuarine species.

The input data comprise three sets:

1. raw point cloud LiDAR data,
2. breaklines, and
3. metadata.

The total number of the tiles is 733. All the results and the maps will be displayed for tile # 37643.

9.4.1. MrSID to LAS

It is possible to convert LiDAR data from MrSID format to LAS format, usinf LizardTech, MG4 Decode version 1.1.0.2801 (Amd64).

The output format is "Decode to LAS" and the output directory is the Marion.LAS folder.

9.4.2. LAS to multipoints

9.4.2.1. Non-ground

The parameters setting list is provided:

The input is the LAS folder.

The output feature class is the non-ground folder.

The average point spacing is 4.

The class code is 1 and 2.

The return value is FIRST RETURN.

The input coordinate system is NAD 1983 HARN State Plane Florida West FIPS 0902 Feet.

The file suffix is las.

The z factor is 1.

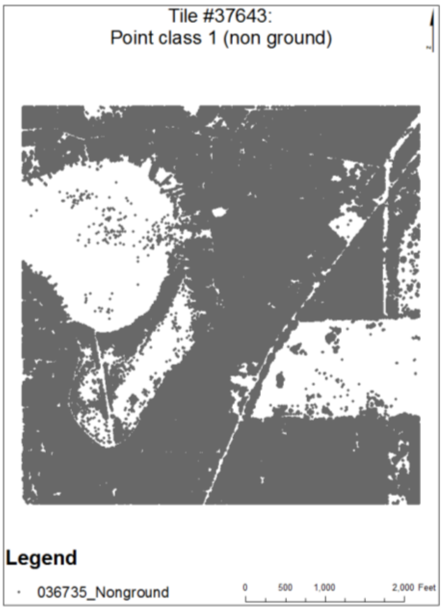


Figure X. Point class 1, tile #37643

9.4.2.2. Ground

The parameters setting list is provided:

The input is the LAS folder.
The output feature class is the ground folder.
The average point spacing is 4.
The class code is 2.
The return value is ANY RETURN.
The input coordinate system is NAD 1983 HARN State Plane Florida West FIPS 0902 Feet.
The file suffix is las.
The z factor is 1.

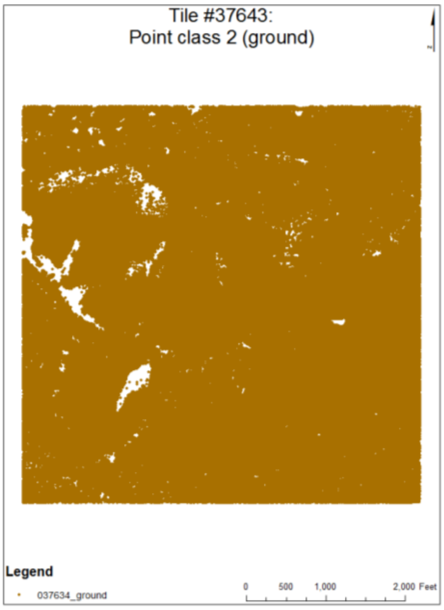


Figure X. Point class 2, tile #37643

9.4.3. Point to raster

9.4.3.1. DSM

The parameters setting list is provided:

The input feature is the non-ground folder.
The value field is Z.
The output feature is the DSM folder.
The cell assignment is the maximum value of the attributes of the points within the cell.
The priority field is none.
The cell size is 5 ft.
The following image shows the DSM, tile #37643.

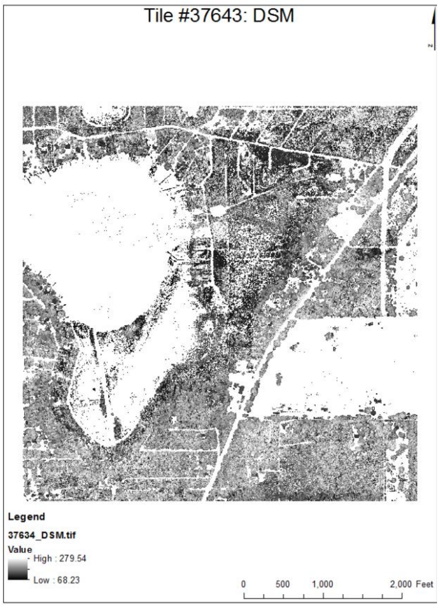


Figure 9.4 DSM, tile #37643

9.4.3.2. DEM

The parameters setting list is provided:

The input feature is the ground folder.
The value field is Z.
The output feature is the DEM folder.
The cell assignment is the maximum value of the attributes of the points within the cell.
The priority field is none.
The cell size is 5 ft.
The following image shows the DEM, tile #37643.

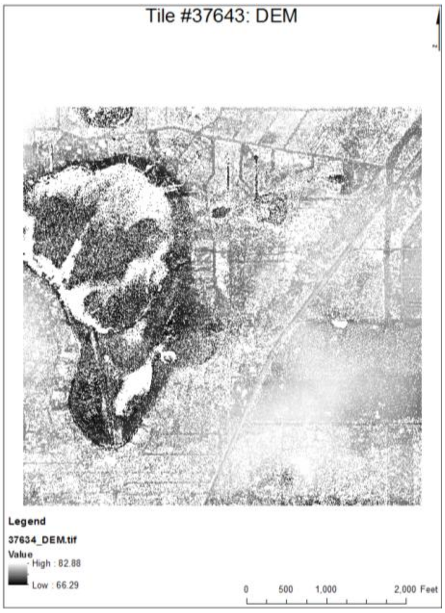


Figure 9.5 DEM, tile #37643

9.4.4. nDSM: DSM - DEM

The following image shows the nDSM, tile #37643.

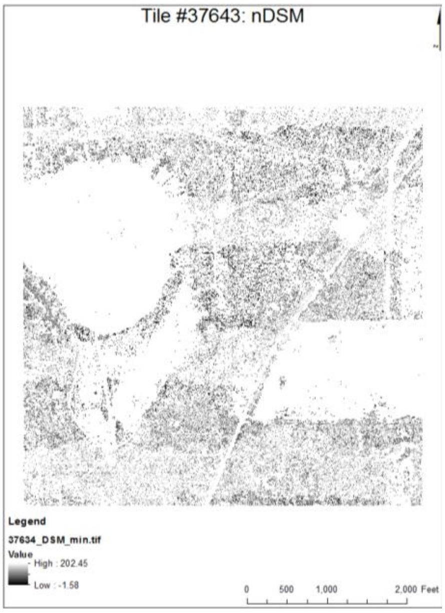


Figure 9.6 DSM - DEM, tile #37643

9.4.5. Raster to point

The parameters setting list is provided:

The input raster is the nDSM folder.
The output point features is the OutputPoints folder.

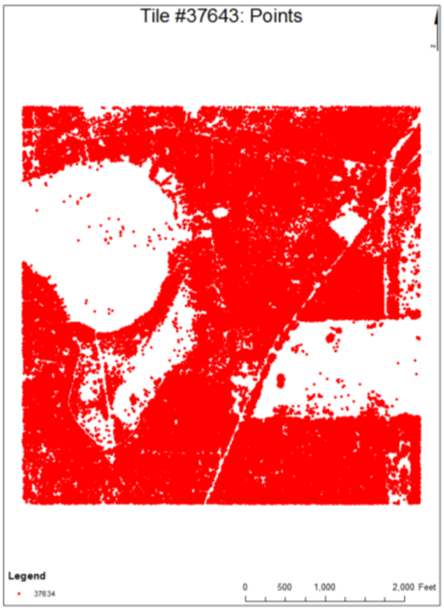


Figure X. Points, tile #37643

9.4.6. Select layer by attribute

9.4.6.1. <u>Points lower than 0 ft</u>

The parameters setting list is provided:

The input folder is the OutputPoints.
The output folder is the LowerThan0.
The selection type is the "NEW_SELECTION".
The where clause is the following: "grid_code <= 0 ft".

The following image shows the points lower than 0 ft, tile #37643.

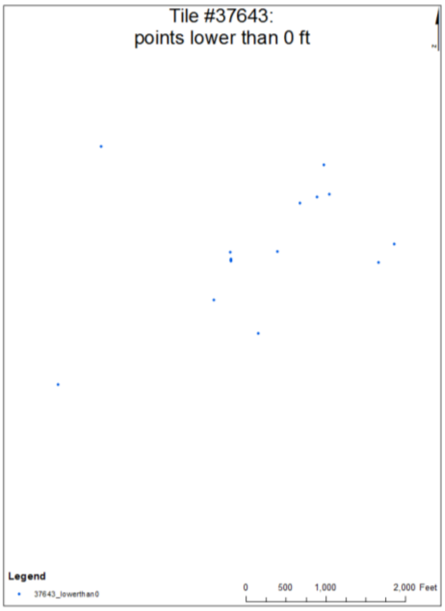


Figure 9.7 Points lower than 0 ft, tile #37643

9.4.6.2. <u>Points between 0 ft and 3.28 ft (1 m)</u>

The parameters setting list is provided:

The input folder is the LowerThan0.
The output folder is the LowerThan3.
The selection type is the "NEW_SELECTION".
The where clause is "grid_code >= 3.28 ft".

The following image shows the points between 0 ft and 3.28 ft, tile #37643.

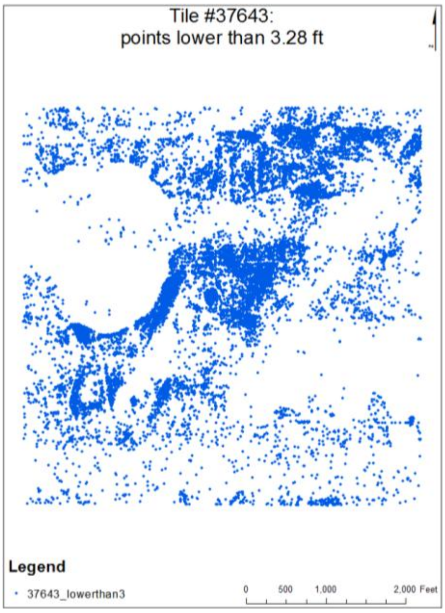


Figure 9.8 Points between 0 ft and 3.28 ft, tile #37643

9.4.6.3. Points between 3.28 ft (1 m) and 6.56 ft (2 m)

The parameters setting list is provided:

The input folder is the LowerThan3.
The output folder is the LowerThan6.
The selection type is the "NEW_SELECTION".
The where clause is "grid_code >= 6.56 ft".

The following image shows the points between 3.28 ft and 6.56 ft, tile #37643.

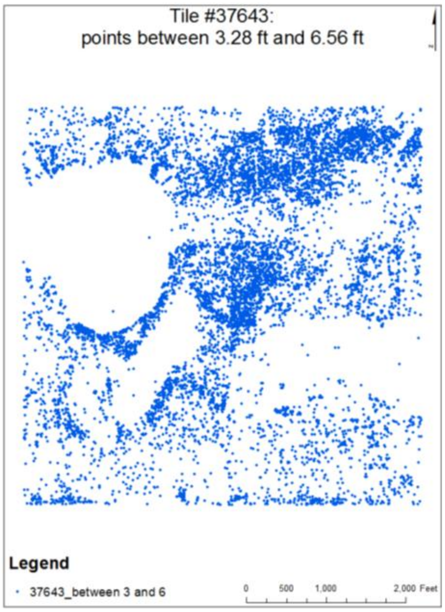


Figure 9.9 Points between 3.28 ft and 6.56 ft, tile #37643

9.4.6.4. Points between 6.56 ft (2 m) and 19.568 ft (6 m)

The parameters setting list is provided:

The input folder is the LowerThan6.
The output folder is the LowerThan19.
The selection type is the "NEW_SELECTION".
The where clause is "grid_code >= 19.658 ft".

The following image shows the points between 6.56 ft and 19.658 ft, tile #37643.

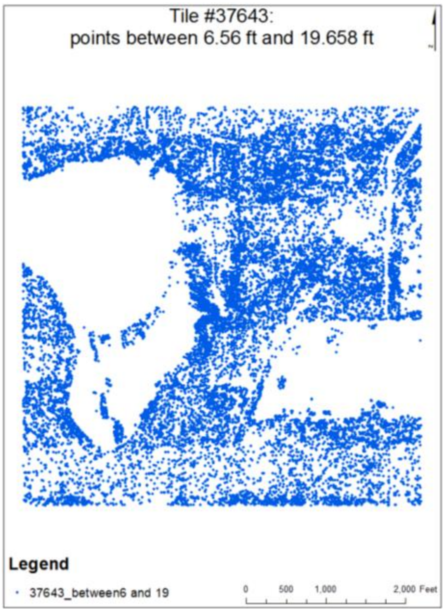


Figure 9.10 Points between 6.56 ft and 19.658 ft, tile #37643

9.4.7. Split

Splitting the Input Features creates a subset of multiple output feature classes.

9.4.7.1. Roads

The parameters setting list is provided:

The input feature is the road breaklines.
The split feature is the footprint breaklines.
The split field is the cell number.
The cluster tolerance is 0 ft.
The output feature is the split roads.

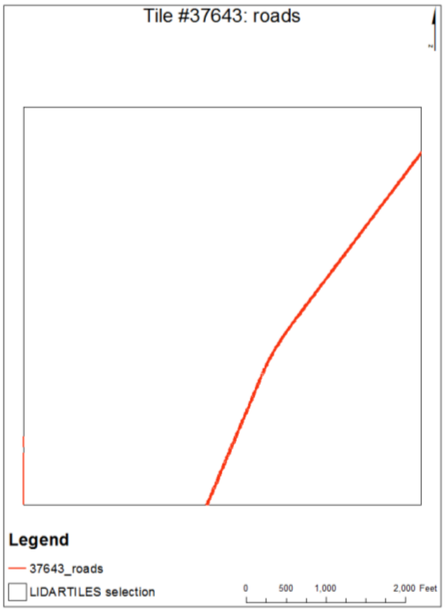


Figure X. Roads, tile #37643

9.4.7.2. Hydrograph

The parameters setting list is provided:

The input feature is the hydrograph breaklines.
The split feature is the footprint breaklines.
The split field is the cell number.
The cluster tolerance is 0 ft.
The output feature is the split hydrograph.

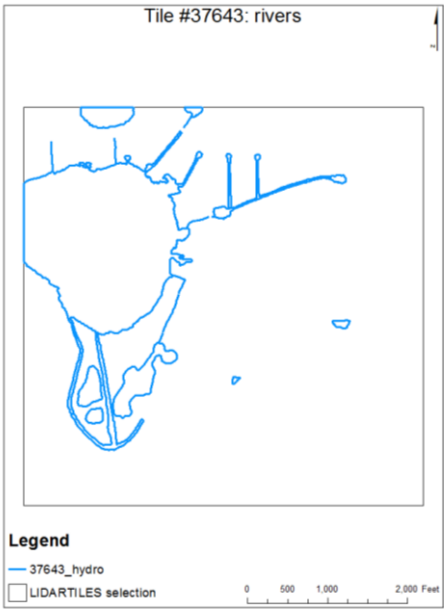


Figure X. Rivers, tile #37643

9.4.8. Feature to polygon

9.4.8.1. Roads

The parameters setting list is provided:

The input feature is the split roads folder.
The output feature class is the ftp_road folder.
The cluster tolerance is 0.2 ft.
There are "NO ATTRIBUTES".

The following image shows the roads in tile #37643.

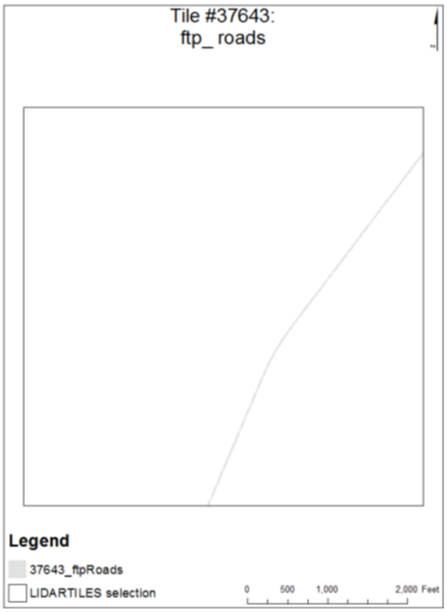


Figure X. ftp_roads, tile #37643

9.4.8.2. Hydrograph

The parameters setting list is provided:

The input feature is the split hydrograph folder.
The output feature class is the ftp_ hydrograph folder.
The cluster tolerance is 0.2 ft.
There are "NO ATTRIBUTES".

The following image shows the rivers in tile #37643.

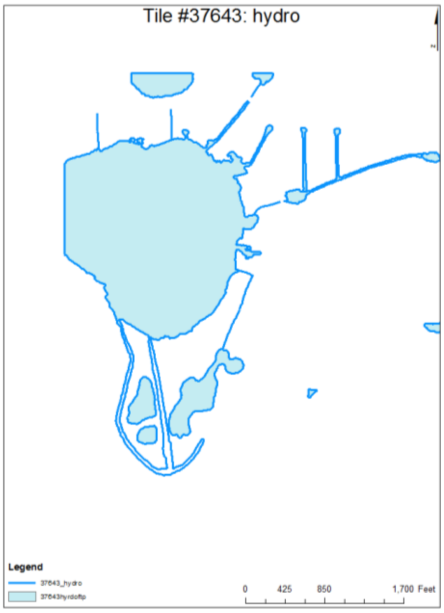


Figure X. ftp_ hydrograph, tile #37643

9.4.9. Clip

9.4.9.1. Water

The input features is the LowerThan3 folder.
The clip features is the split hydros.
The output feature class is waterpoints
The cluster tolerance is 0 ft.

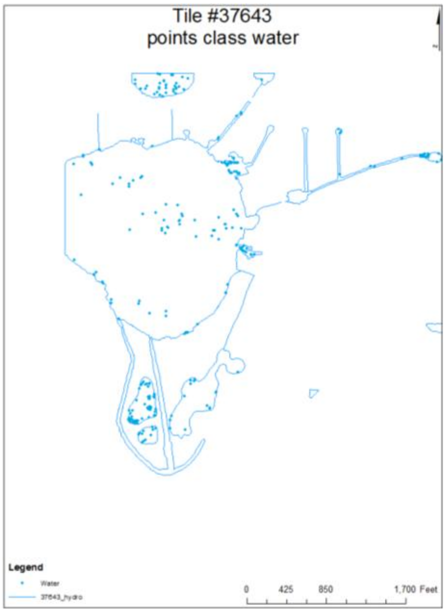


Figure X. Points classified as river 0 - 3.28 ft, tile #3764

9.4.10. Select by Location

9.4.10.1. <u>Vegetation between 0 ft and 3.28 ft</u>

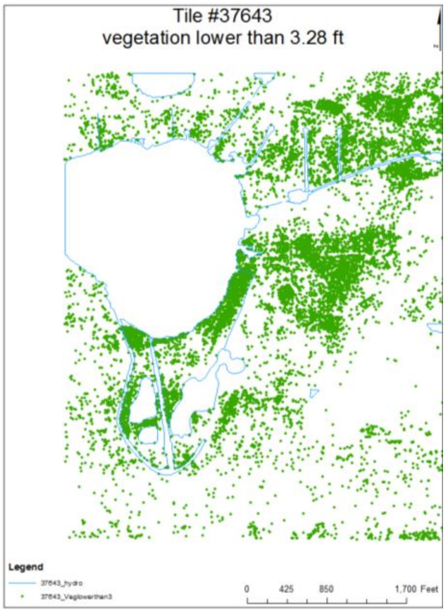


Figure X. Points classified as vegetation 0 - 3.28 ft, tile #3764

PART III

APPENDIX A

Softwares and site-package

The aim of this dissertation is to obtain 3D models in an automatic way. For this reason, softwares and site-package are used. The following list provides the ones used:

- Fortran (release 95),
- Matlab (release 12),
- ArcGIS(release 10),
- Model builder,
- ArcPy, and
- Visual Nature Studio (release 3).

The following sections describe the different softwares and site-package used.

A.1.FORTRAN

Originally it was developed by IBM at their campus in south San Jose, California in 1957 for scientific and engineering applications. The name FORTRAN is an acronym for FORmula TRANslation.

Because it is designed to allow easy translation of math formulas into code, it has been in continual use for over half a century in computationally intensive areas such as numerical weather prediction, finite element analysis, computational fluid dynamics, computational physics and computational chemistry.

The first standardized version has come to be known as FORTRAN '66 (also known as FORTRAN IV). The second was FORTRAN 77. It was released in 1978 (it was called '77 because the Association began it is review in 1977). Fortran 90 was released in 1990, using the new capitalization scheme. Fortran 95 added functional programming. Fortran 2003 added object-oriented programming and generic programming. Fortran 2008 added concurrent programming.

FORTRAN was the first high-level language, using the first compiler ever developed. The objective during its design was to create a programming language that would be:

- simple to learn,
- suitable for a wide variety of applications,
- machine independent,
- would allow complex mathematical expressions to be stated similarly to regular algebraic notation and
- still being almost as efficient in execution as assembly language.

A.2. MATLAB

Cleve Moler, the chairman of the computer-science department at the University of New Mexico, started developing MATLAB in the late 1970s.

The name MATLAB comes from MATrix LABoratory. Matrix because the data type used is the matrix, laboratory because MATLAB is used at school for didactics and research. MATLAB allows matrix manipulations, plotting of data, implementation of algorithms, and interfacing with programs written in other languages, including C, C++, Java, and Fortran.

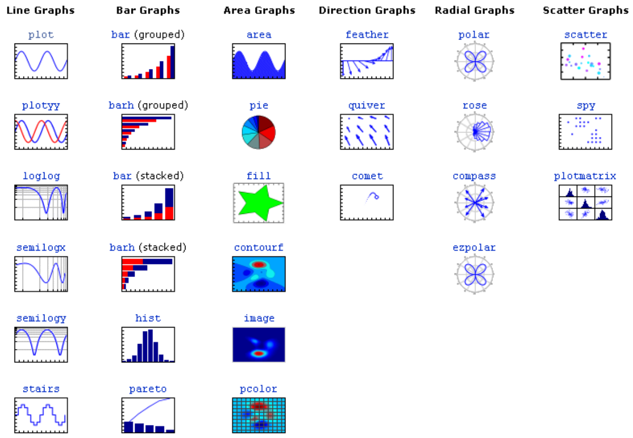
A.2.1. Graphics

MATLAB is very useful for making plots in an automatic way.

MATLAB offers a variety of data plotting functions to create graphic displays. Plots can display data and annotations such as titles, legends, and colorbars.

There are different types of MATLAB plots: two-dimensional plotting functions and three-dimensional plotting functions.

This table shows MATLAB 2D plotting functions.



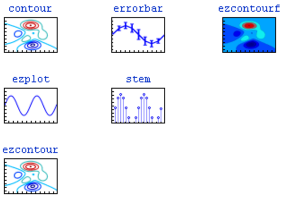


Figure X. Different types of 2D plots

The following table shows MATLAB 3D and volumetric plotting functions.

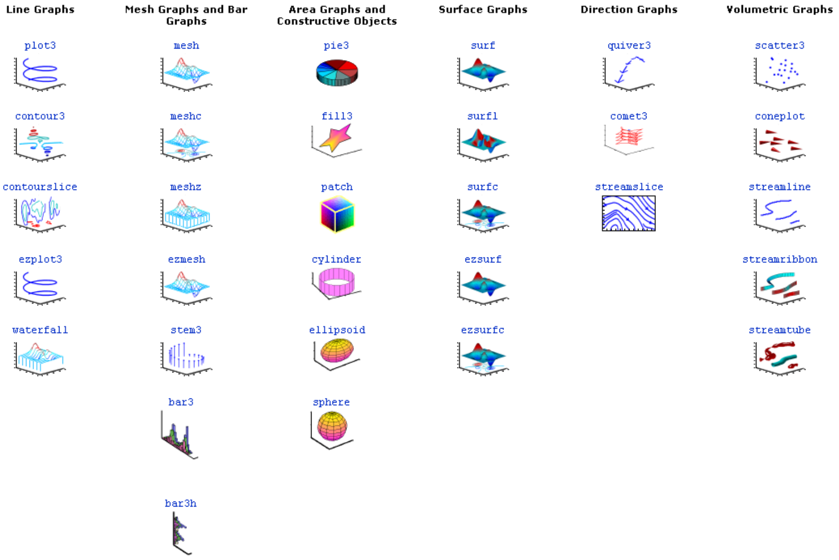


Figure X. Different types of 3D plots

A.2.2. Triangulation

It displays each triangle defined in the m-by-3 matrix TRI as a surface in 3D space.

The command TRI = delaunay(X,Y) creates a 2D Delaunay triangulation of the points (X,Y), where X and Y are column-vectors. TRI is a matrix representing the set of triangles that make up the triangulation. The matrix is of size mtri-by-3, where mtri is the number of triangles. Each row of TRI specifies a triangle defined by indices with respect to the points.

Another way of displaying data in a 3D space is using the command Trisurf (tri,x,y,z). It displays each triangle defined in the m-by-3 matrix TRI as a surface in 3D space.

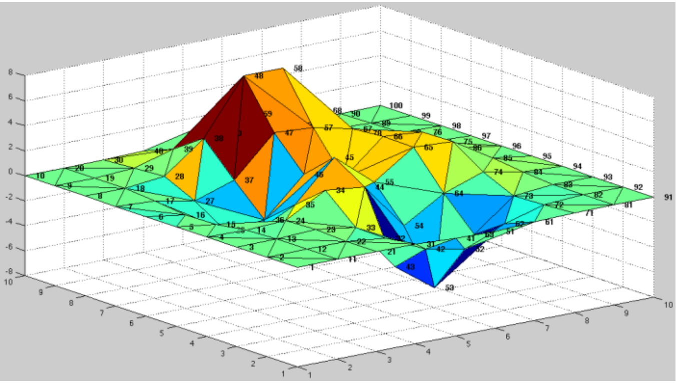


Figure x. MATLAB Delaunay triangulation

A.3. MODEL BUILDER

Model builder is a way to automate a series of tools. It is a part of the ArcGIS geoprocessing framework. A framework is a mechanics of using, managing, and publishing tools. It can automate workflows by creating new tools, models and scripts.
A model comprises different elements:

- input value (also called workspace),
- a tool, and
- a derived value.

The input data are the data provided to the model. The derived data are created by the tools in the model. The input data, the tool and the derived data make the process. Each process passes through three steps:

- not-ready to run,
- ready-to run, and
- has-been-run.

Figure x shows a model not-ready to run.

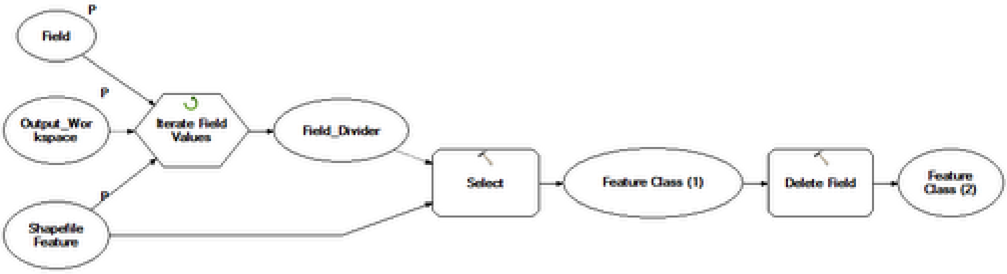


Figure x. Not-ready to run model

Figure x shows a model read-to run.

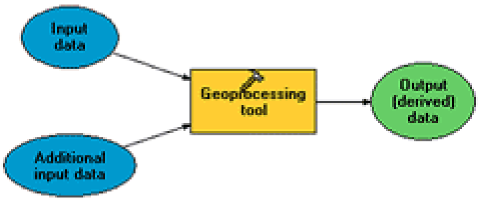


Figure x. Ready-to run model

It is possible to iterate a process with iterators. The following picture shows a model with the input data in a blue elliptical shape, the iterator in a orange hexagon shape and the output in a green elliptical shape.

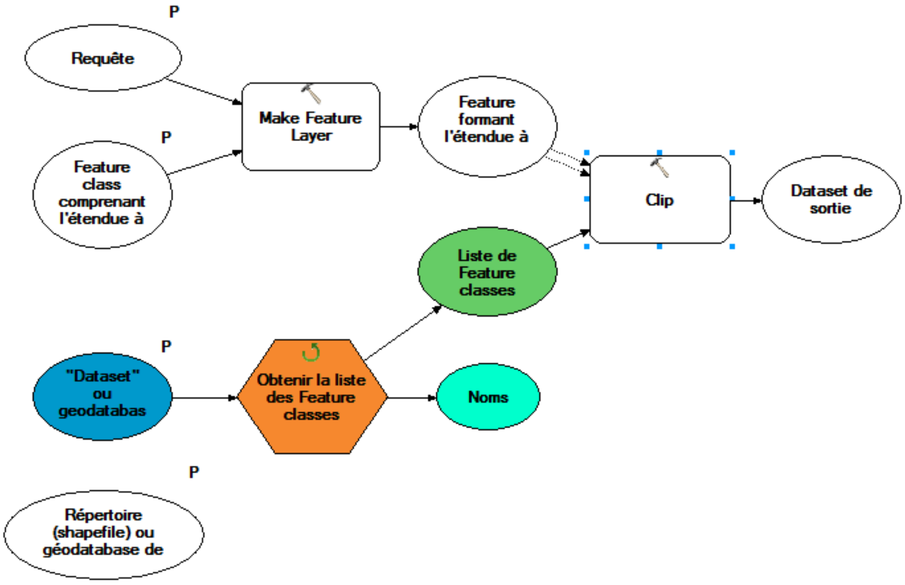

P
Requête
P
Feature class comprenant l'étendue à
Make Feature Layer
Feature formant l'étendue à
Clip
Dataset de sortie
Liste de Feature classes
P
"Dataset" ou geodatabas
Obtenir la liste des Feature classes
Noms
P
Répertoire (shapefile) ou géodatabase de

Figure x. An example of a model

They run the entire model a number of times based on the properties of the iterator. Only one iterator is allowed per a model, but is possible to nest models to combine iterators. The input is a workspace and for every feature class in this workspace, this model will run. The iterator will run the entire model once so it is a for loop.

Two different ModelBuilder iterators will be used:

- feature classes: it works for each feature class in a workspace and
- files: it works for each file in a folder.

The outputs of this type of iterator are two: the former is the output feature class and the last one is the output name.

A.4. ARCPY

A.4.1. What is Python?

Python is a free, cross-platform, open-source programming language. Python was introduced to the ArcGIS community at 9.0. Since then, it has been accepted as the scripting language of choice for geoprocessing users and continues to grow.

Here are just some of the advantages of Python:

- easy to learn and excellent for beginners, yet superb for experts,
- highly scalable, suitable for large projects or small one-off programs known as scripts,
- portable, cross-platform,

- embeddable (making ArcGIS scriptable),
- stable and mature, and
- a large user community.

Python extends across ArcGIS and becomes the language for data analysis, data conversion, data management, and map automation, helping increase productivity.

A.4.2. What is ArcPy?

ArcGIS 10 introduces ArcPy. It develops Python scripts while offering code completion and integrated documentation for each function, module, and class.

The power of using ArcPy within Python is the fact that Python is a general purpose programming language. Python script tools can embed the script in the toolbox. In this way, only the toolbox is needed.

ArcPy is a site package that builds on the successful ArcGIS scripting module. Its goal is to create the cornerstone for a useful and productive way to perform geographic data analysis, data conversion, data management, and map automation with Python.

This package provides a rich and native Python experience offering code completion and reference documentation for each function, module, and class.

The additional power of using ArcPy within Python is the fact that Python is a general-purpose programming language.

A.4.3. Essential ArcPy vocabulary

This sub-section introduces some terms to better understanding the ArcPy help.

The terms are:

- ArcPy modules,
- ArcPy classes, and
- ArcPy functions.

The definitions are the following ones:

- ArcPy modules: a module is a python file that generally includes functions and classes. ArcPy is supported by a series of modules, including a mapping module (arcpy.mapping), a Spatial Analyst module (arcpy.sa), and a Geostatistical Analyst module (arcpy.ga). [ArcGIS resource Center].

- ArcPy classes: a class is analogous to an architectural blueprint. The blueprint provides the framework for how to create something. Classes can be used to create objects. ArcPy classes, such as the SpatialReference and Extent classes, are often used as shortcuts to complete geoprocessing tool parameters that would otherwise have a more complicated string equivalent. [ArcGIS resource Center].

- ArcPy functions: a function is a defined bit of functionality that does a specific task and can be incorporated into a larger program. In ArcPy all geoprocessing tools are provided as functions, but not all functions are geoprocessing tools. In addition to tools, ArcPy provides a number of functions to better support geoprocessing Python workflows. Functions or methods can be used to list certain datasets, retrieve a dataset's properties, validate a table name before adding it to a geodatabase, or perform many other useful scripting tasks. [ArcGIS resource Center]

A.5. VISUAL NATURE STUDIO (VNS)

Visual Nature Studio 3 makes creating complex scenes and faster rendering. The users of Visual Nature Studio (VNS) include foresters, engineers, landscape architects, GIS professionals, historians, graphic artists and many, many others.

It is possible to import the GIS data and make a photorealistic image that anyone can easily understand. VNS provides tools to control visualization directly from GIS data, simplifying and automating the process.

VNS offers easy ways to landscape the scene whether it involves growing a lawn, landscaping a subdivision or growing and harvesting entire forests. Add roads, lakes, streams through the use of landform modifying capabilities.

APPENDIX B

BÉZIER SPLINES

The Bézier spline is used, because it forms continuous surfaces with one continuous derivative surfaces.
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.

Polynomials can be pieced together to form spline curves that can approximate any function to any accuracy desired. In general, any polynomial function that has degree less than or equal to n, can be written in the following way:

$$p(t) = a_n t^n + a_{n-1} t^{n-1} + \cdots + a_1 t + a_0$$

The set of polynomials of degree less than or equal to n forms a vector space: polynomials can be added together, can be multiplied by a scalar, and all the vector space properties hold.
The set of functions $\{1, t, t^2, \dots, t^n\}$ form a basis for this vector space – that is, any polynomial of
degree less than or equal to n can be uniquely written as a linear combinations of these functions. This basis, commonly called the power basis, is only one of an infinite number of bases for the space of polynomials. Another commonly used bases for the space of polynomials are the Bernstein basis.

The Bernstein polynomials of degree n are defined by:

$$B_{i,n}(t) = \sum_{i=0}^{n} \binom{n}{i} P_i (1-t)^{n-i} t^i \, , t \in [0,1]$$

$$B_{i,n}(t) = P_0 (1-t)^n + \binom{n}{1} P_1 (1-t)^{n-1} t + \cdots + \binom{n}{n} P_n t^n, \qquad t \in [0,1]$$

The polynomials $\binom{n}{i}(1-t)^{n-i}t^i$ are known as Bernstein basis polynomials of degree n and they are defined by $b_{i,n}(t) \coloneqq \binom{n}{i}(1-t)^{n-i}t^i, i = 0, \dots, n$.

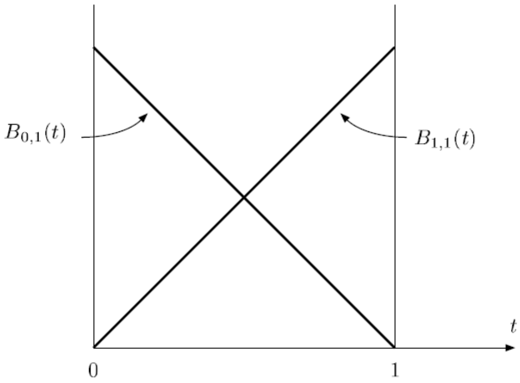
$$\binom{n}{i} = \frac{n!}{i!\,(n-i)!}$$

the coefficients $\binom{n}{i}$ can be obtained from Pascal's triangle; the exponents on the t term increase by one as i increases; and the exponents on the $(1 - t)$ term decrease by one as i increases.

The Bernstein polynomials of degree 1 are:

$$B_{0,1}(t) = 1 - t$$
$$B_{1,1}(t) = t$$

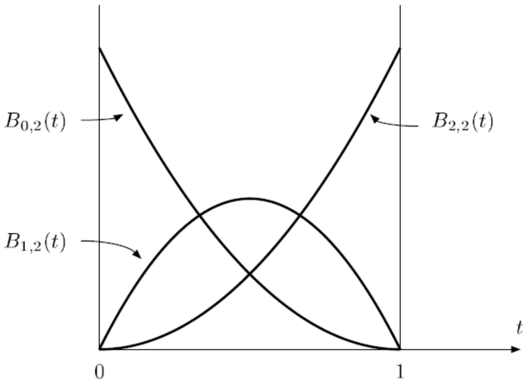
and can be plotted for $0 \le t \le 1$ as



The Bernstein polynomials of degree 2 are:

$$B_{0,2}(t) = (1 - t)^2$$
$$B_{1,2}(t) = 2t * (1 - t)$$
$$B_{2,2}(t) = t^2$$

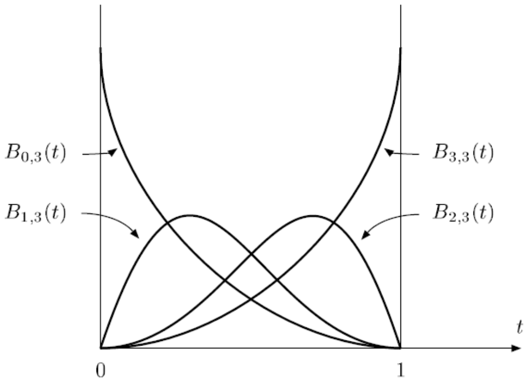
and can be plotted for $0 \le t \le 1$ as:



The Bernstein polynomials of degree 3 are

$$B_{0,3}(t) = (1 - t)^3$$
$$B_{1,3}(t) = 3t * (1 - t)^2$$
$$B_{2,3}(t) = 3t^2 * (1 - t)$$
$$B_{3,3}(t) = t^3$$

and can be plotted for $0 \leq t \leq 1$ as



The following are examples of different Bézier splines.

B.1. Linear Bézier curve

Given points P_0 and P_1, a linear Bézier curve is simply a straight line between those two points.

$$P_0 = 0;0$$
$$P_1 = 6;5$$

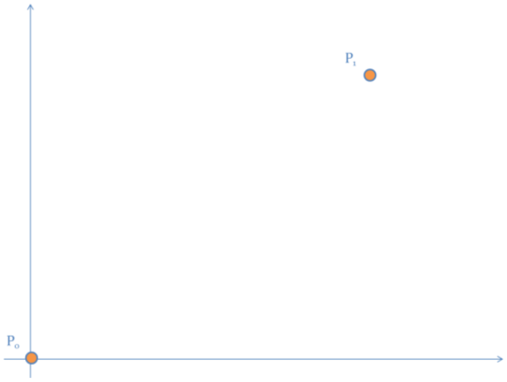


Figure B. Two input points

The curve is given by

$$B(t) = \sum_{i=0}^{n} \binom{n}{i} P_i (1-t)^{n-i} t^i \, , t \in [0,1]$$

and is equivalent to linear interpolation.
The explicit form of the curve is:

$$B(t) = \sum_{i=0}^{1} \binom{1}{i} P_i (1-t)^{1-i} t^i =$$
$$B(t) = \binom{1}{0} P_0 (1-t)^{1-0} t^0 + \binom{1}{1} P_1 (1-t)^{1-1} t =$$
$$B(t) = \frac{1!}{0!\,(1-0)!} P_0 (1-t) + \frac{0!}{0!\,(0-0)!} P_1 t =$$

$$B(t) = P_0 (1-t) + P_1 t$$

$$x_{B(t)} = (1 - t) * x_{P_0} + t * x_{P_1}$$
$$y_{B(t)} = (1 - t) * y_{P_0} + t * y_{P_1}$$

$$x_{B(0)} = (1 - 0) * 0 + 0 * 6 = 0$$
$$y_{B(0)} = (1 - 0) * 0 + 0 * 5 = 0$$
$$P^I = 0; 0$$

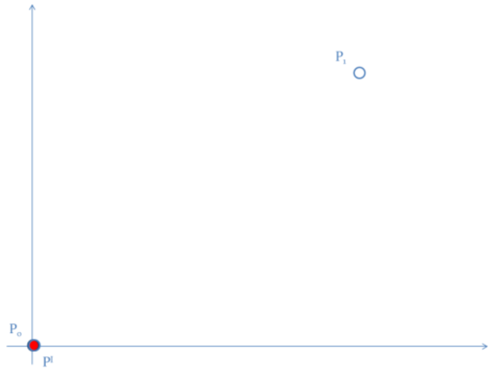


Figure B. Point P^{I}

$$x_{B(0.25)} = (1 - 0.25) * 0 + 0.25 * 6 = 1.5$$
$$y_{B(0.25)} = (1 - 0.25) * 0 + 0.25 * 5 = 1.25$$
$$P^{II} = 1.5; 1.25$$

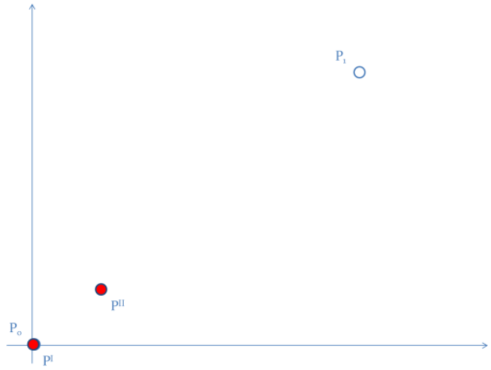


Figure B. Points P^{I} and P^{II}

$$x_{B(0.5)} = (1 - 0.5) * 0 + 0.5 * 6 = 3$$
$$y_{B(0.5)} = (1 - 0.5) * 0 + 0.5 * 5 = 2.5$$
$$P^{III} = 3; 2.5$$

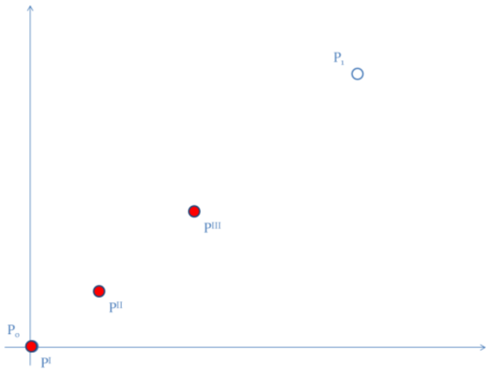


Figure B. Points P^I , P^{II} and P^{III}

$$x_{B(0.75)} = (1 - 0.75) * 0 + 0.75 * 6 = 4.5$$
$$y_{B(0.75)} = (1 - 0.75) * 0 + 0.75 * 5 = 3.75$$
$$P^{IV} = 4.5; 3.75$$

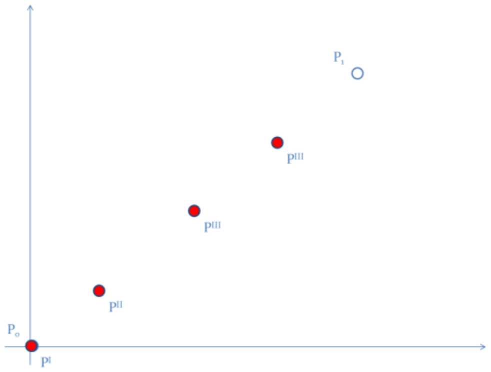


Figure B. Points P^I , P^{II}, P^{III} and P^{IV}

$$x_{B(1)} = (1 - 1) * 0 + 1 * 6 = 6$$
$$y_{B(1)} = (1 - 1) * 0 + 1 * 5 = 5$$
$$P^V = 6; 5$$

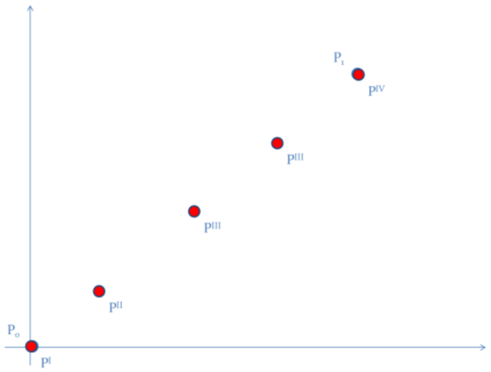


Figure B. Points P^I , P^{II}, P^{III}, P^{IV} and P^V

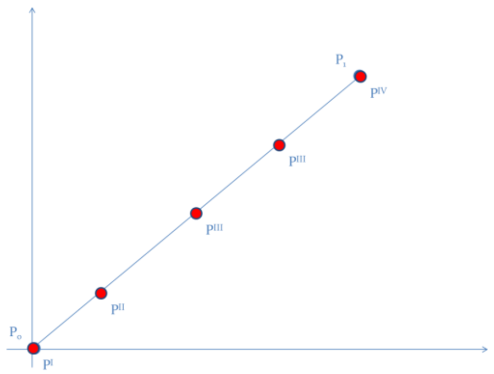


Figure B. Points P^{I}, P^{II}, P^{III}, P^{IV} and P^{V}, Bézier spline (in light blue) and straight line (orange)

La retta e la spline si sovrappongono.

B.2. Quadric Bézier curve

A quadratic Bézier curve is the path traced by the function B(t), given points P_0, P_1, and P_2.

$$P_0 = 0;0$$
$$P_1 = 0;5$$
$$P_2 = 6;5$$

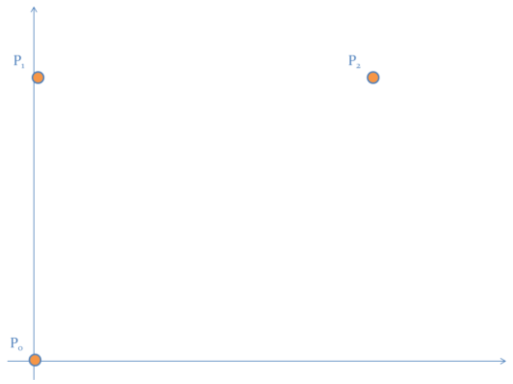


Figure B. Three input points

The general form of the curve is:

$$B(t) = \sum_{i=0}^{n} \binom{n}{i} P_i (1-t)^{n-i} t^i \, , t \in [0,1]$$

It can be interpreted as the linear interpolation of corresponding points on the linear Bézier curves from P_0 to P_1 and from P_1 to P_2 respectively.

The explicit form of the curve is:

$$B(t) = \sum_{i=0}^{2} \binom{2}{i} P_i (1-t)^{2-i} t^i =$$
$$B(t) = \binom{2}{0} P_0 (1-t)^2 t^0 + \binom{2}{1} P_1 (1-t)^{2-1} t + \binom{2}{2} P_2 (1-t)^{3-2} t^2 =$$
$$B(t) = \frac{2!}{0!\,(2-0)!} P_0 (1-t)^2 + \frac{2!}{1!\,(2-1)!} P_1 (1-t) t + \frac{2!}{2!\,(2-2)!} P_2 t^2 =$$

$$B(t) = P_0(1-t)^2 + 2P_1(1-t)t + P_2t^2$$

$$x_{B(t)} = (1-t^2) * x_{P_0} + 2 * t * x_{P_1} + t^2 * x_{P_2}$$
$$y_{B(t)} = (1-t^2) * y_{P_0} + 2 * t * y_{P_1} + t^2 * y_{P_2}$$

$$x_{B(0)} = (1-0^2) * 0 + 2 * 0 * 0 + 0^2 * 6 = 0$$
$$y_{B(0)} = (1-0^2) * 0 + 2 * 0 * 5 + 0^2 * 5 = 0$$
$$P^I = 0; 0$$

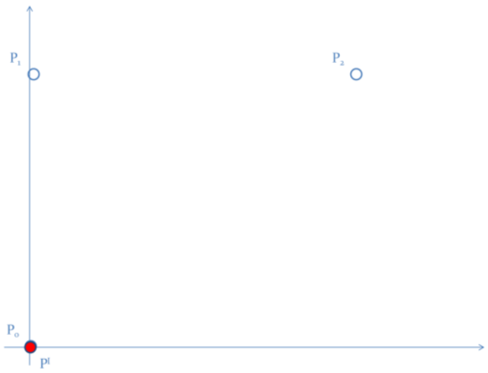


Figure B. Point P^I

$$x_{B(0.25)} = (1-0.25^2) * 0 + 2 * 0.25 * 0 + 0.25^2 * 6 = 0.375$$
$$y_{B(0.25)} = (1-0.25^2) * 0 + 2 * 0.25 * 5 + 0.25^2 * 5 = 2.1875$$
$$P^{II} = 0.375; 2.1875$$

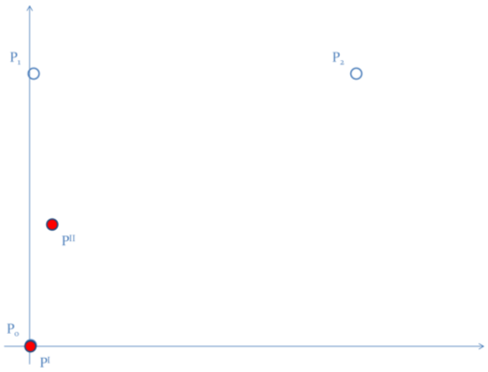


Figure B. Points P^I and P^{II}

$$x_{B(0.5)} = (1-0.5^2) * 0 + 2 * 0.5 * 0 + 0.5^2 * 6 = 1.5$$
$$y_{B(0.5)} = (1-0.5^2) * 0 + 2 * 0.5 * 5 + 0.5^2 * 5 = 3.75$$
$$P^{III} = 1.5; 3.75$$

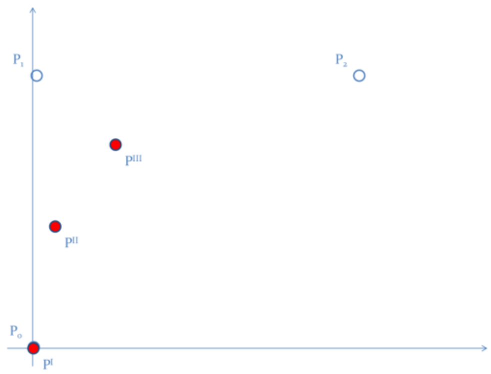


Figure B. Points P^{I}, P^{II} and P^{III}

$$x_{B(0.75)} = (1 - 0.75^2) * 0 + 2 * 0.75 * 0 + 0.75^2 * 6 = 3.375$$
$$y_{B(0.75)} = (1 - 0.75^2) * 0 + 2 * 0.75 * 5 + 0.75^2 * 5 = 4.6875$$
$$P^{IV} = 3.375; 4.6875$$

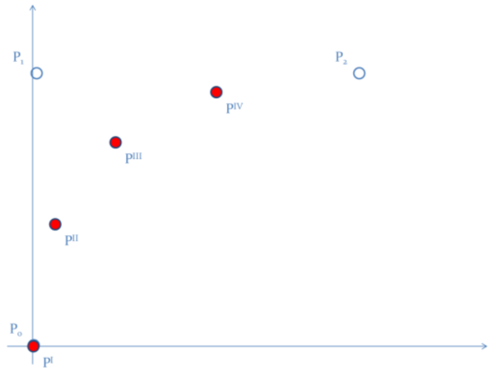


Figure B. Points P^{I}, P^{II}, P^{III} and P^{IV}

$$x_{B(1)} = (1 - 1^2) * 0 + 2 * 1 * 0 + 1^2 * 6 = 6$$
$$y_{B(1)} = (1 - 1^2) * 0 + 2 * 1 * 5 + 1^2 * 5 = 5$$
$$P^{V} = 6; 5$$

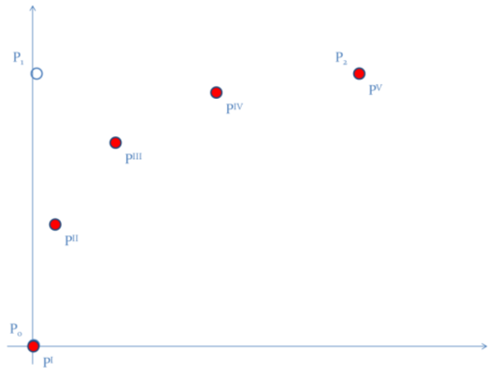


Figure B. Points P^{I}, P^{II}, P^{III}, P^{IV} and P^{V}

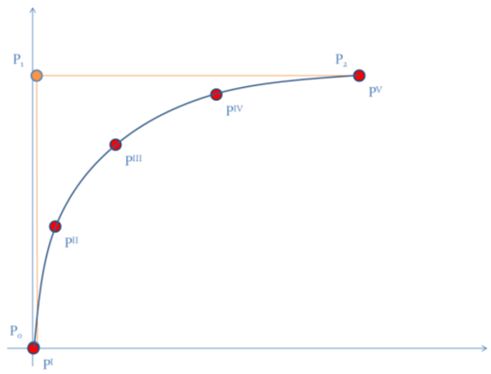


Figure B. Points P^{I}, P^{II}, P^{III}, P^{IV} and P^{V}, Bézier spline (in light blue) and straight line (orange)

It departs from P_0 in the direction of P_1, then bends to arrive at P_2 in the direction from P1. In other words, the tangents in P_0 and P_2 both pass through P_1.

A quadratic Bézier curve is also a parabolic segment. As a parabola is a conic section, some sources refer to quadratic Béziers as "conic arcs".

B.3. Cubic Bézier curve

Four points P_0, P_1, P_2 and P_3 in the plane or in higher-dimensional space define a cubic Bézier curve. The curve starts at P_0 going toward P_1 and arrives at P_3 coming from the direction of P_2. Usually, it will not pass through P_1 or P_2; these points are only there to provide directional information. The distance between P_0 and P_1 determines "how long" the curve moves into direction P_2 before turning towards P_3.

$$P_0 = 0;0$$
$$P_1 = 0;5$$
$$P_2 = 5;5$$
$$P_3 = 6;2$$

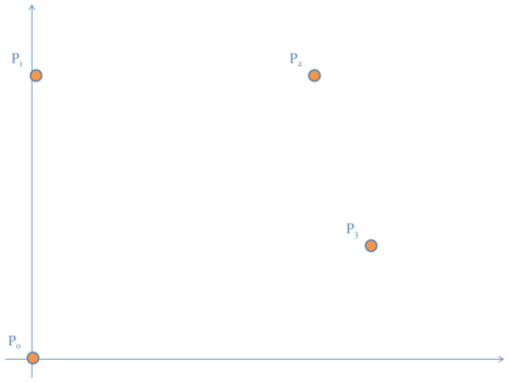


Figure B. Four input points

The general form of the curve is:

$$B(t) = \sum_{i=0}^{n} \binom{n}{i} P_i (1-t)^{n-i} t^i \, , t \in [0{,}1]$$

The explicit form of the curve is:

$$B(t) = \sum_{i=0}^{3} \binom{3}{i} P_i (1-t)^{3-i} t^i$$

$$B(t) = \binom{3}{0} P_0 (1-t)^3 t^0 + \binom{3}{1} P_1 (1-t)^{3-1} t + \binom{3}{2} P_2 (1-t)^{3-2} t^2 + \binom{3}{3} P_3 (1-t)^{3-3} t^3 =$$

$$B(t) = \frac{3!}{0!\,(3-0)!}P_0(1-t)^3 + \frac{3!}{1!\,(3-1)!}P_1(1-t)^2t + \frac{3!}{2!\,(3-2)!}P_2(1-t)t^2 + \frac{3!}{3!\,(3-3)!}P_3t^3 =$$
$$B(t) = P_0(1-t)^3 + 3P_1(1-t)^2t + 3P_2(1-t)t^2 + P_3t^3$$

$$x_{B(t)} = (1-t)^3 * x_{P_0} + 3 * t * (1-t)^2 * x_{P_1} + 3 * t^2 * (1-t) * x_{P_2} + x_{P_3} * t^3$$
$$y_{B(t)} = (1-t)^3 * y_{P_0} + 3 * t * (1-t)^2 * y_{P_1} + 3 * t^2 * (1-t) * y_{P_2} + y_{P_3} * t^3$$

$$x_{B(0)} = (1-0)^3 * 0 + 3 * 0 * (1-0)^2 * 0 + 3 * 0^2 * (1-0) * 5 + 6 * 0^3 = 0$$
$$y_{B(0)} = (1-0)^3 * 0 + 3 * 0 * (1-0)^2 * 5 + 3 * 0^2 * (1-0) * 5 + 2 * 0^3 = 0$$
$$P^I = 0; 0$$

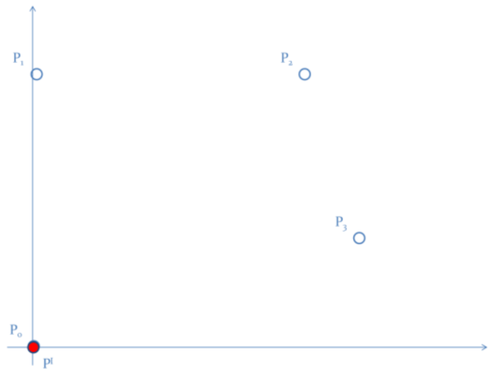


Figure B. Point P^I

$$x_{B(0.25)} = (1-0.25)^3 * 0 + 3 * 0.25 * (1-0.25)^2 * 0 + 3 * 0.25^2 * (1-0.25) * 5 + 6 * 0.25^3 = 0.7969$$
$$y_{B(0.25)} = (1-0.25)^3 * 0 + 3 * 0.25 * (1-0.25)^2 * 5 + 3 * 0.25^2 * (1-0.25) * 5 + 2 * 0.25^3 = 2.8434$$
$$P^{II} = 0.7969; 2.8434$$

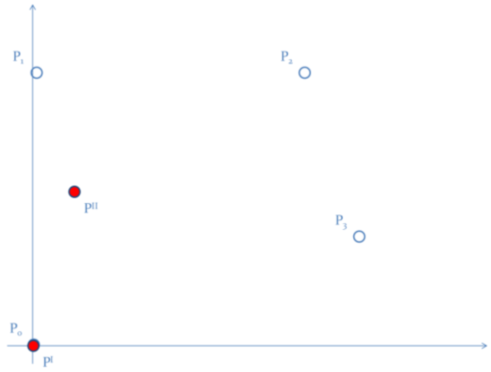


Figure B. Points P^I and P^II

$$x_{B(0.5)} = (1-0.5)^3 * 0 + 3 * 0.5 * (1-0.5)^2 * 0 + 3 * 0.5^2 * (1-0.5) * 5 + 6 * 0.5^3 = 2.625$$
$$y_{B(0.5)} = (1-0.5)^3 * 0 + 3 * 0.5 * (1-0.5)^2 * 5 + 3 * 0.5^2 * (1-0.5) * 5 + 2 * 0.5^3 = 4$$
$$P^{III} = 2.625; 4$$

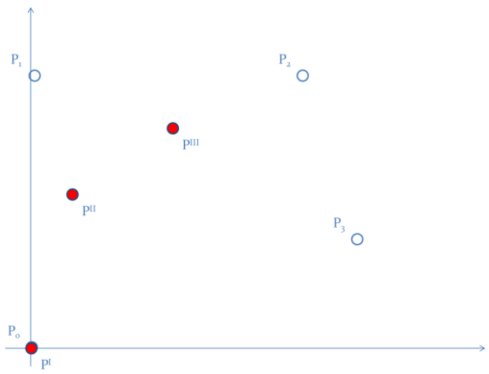


Figure B. Points P^{I}, P^{II} and P^{III}

$$x_{B(0.75)} = (1-0.75)^3 * 0 + 3 * 0.75 * (1-0.75)^2 * 0 + 3 * 0.75^2 * (1-0.75) * 5 + 6 * 0.75^3 = 4.6406$$
$$y_{B(0.75)} = (1-0.75)^3 * 0 + 3 * 0.75 * (1-0.75)^2 * 5 + 3 * 0.75^2 * (1-0.75) * 5 + 2 * 0.75^3 = 3.6563$$
$$P^{IV} = 4.6406; 3.6563$$

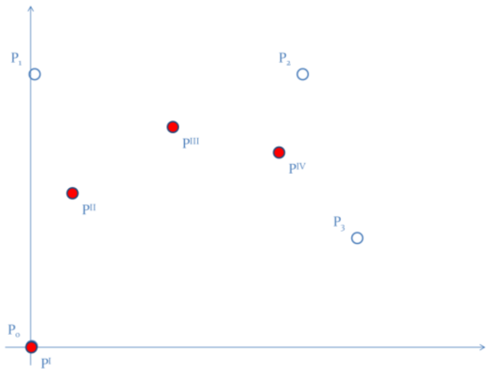


Figure B. Points P^{I}, P^{II}, P^{III} and P^{IV}

$$x_{B(1)} = (1-1)^3 * 0 + 3 * 1 * (1-1)^2 * 0 + 3 * 1^2 * (1-1) * 5 + 6 * 1^3 = 6$$
$$y_{B(1)} = (1-1)^3 * 0 + 3 * 1 * (1-1)^2 * 5 + 3 * 1^2 * (1-1) * 5 + 2 * 1^3 = 2$$
$$P^{V} = 6; 2$$

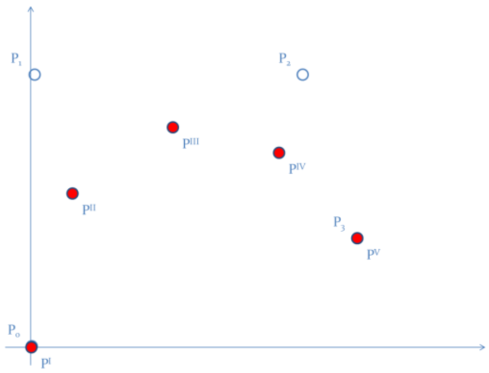


Figure B. Points P^{I}, P^{II}, P^{III}, P^{IV} and P^{V}

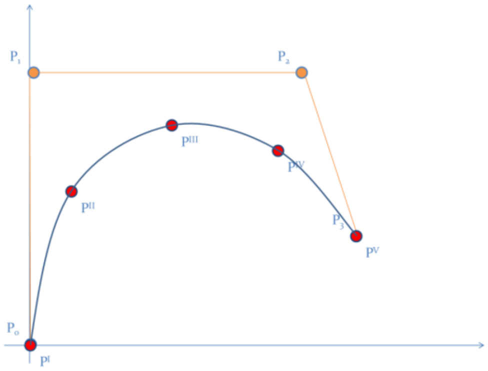


Figure B. Points P^{I}, P^{II}, P^{III}, P^{IV} and P^{V}, Bézier spline (in light blue) and straight line (orange)

B.4. **Fourth-order curves**

$$P_0 = 0; 0$$
$$P_1 = 0; 8$$
$$P_2 = 8; 8$$
$$P_3 = 6; 3$$
$$P_4 = 4; 0$$

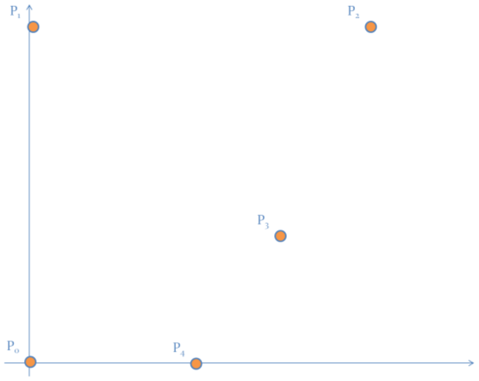


Figure B. Five input points

The general form of the curve is:

$$B(t) = \sum_{i=0}^{n} \binom{n}{i} P_i(1-t)^{n-i}t^i \, , t \in [0,1]$$

The explicit form of the curve is:

$$B(t) = \sum_{i=0}^{4} \binom{4}{i} P_i(1-t)^{4-i}t^i =$$

$$B(t) = \binom{4}{0} P_0(1-t)^4 t^0 + \binom{4}{1} P_1(1-t)^{4-1} t + \binom{4}{2} P_2(1-t)^{4-2} t^2 + \binom{4}{3} P_3(1-t)^{4-3} t^3 + \binom{4}{4} P_4(1-t)^{4-4} t^4 =$$

$$B(t) = \frac{4!}{0!\,(4-0)!} P_0(1-t)^4 + \frac{4!}{1!\,(4-1)!} P_1(1-t)^3 t + \frac{4!}{2!\,(4-2)!} P_2(1-t)^2 t^2 + \frac{4!}{3!\,(4-3)!} P_3(1-t) t^3 + \frac{4!}{4!\,(4-4)!} P_4 t^4 =$$

$$B(t) = P_0(1-t)^4 + 4P_1(1-t)^3t + 6P_2(1-t)^2t^2 + 4P_3(1-t)t^3 + P_4t^4$$

$$x_{B(t)} = (1-t)^4 * x_{P_0} + 4 * t * (1-t)^3 * x_{P_1} + 6 * t^2 * (1-t)^2 * x_{P_2} + 4 * x_{P_3} * (1-t) * t^3 + y_{P_4} * t^4$$
$$y_{B(t)} = (1-t)^4 * y_{P_0} + 4 * t * (1-t)^3 * y_{P_1} + 6 * t^2 * (1-t)^2 * y_{P_2} + 4 * y_{P_3} * (1-t) * t^3 + y_{P_4} * t^4$$

$$x_{B(0)} = (1-0)^4 * 0 + 4 * 0 * (1-0)^3 * 0 + 6 * 0^2 * (1-0)^2 * 8 + 4 * 6 * (1-0) * 0^3 + 4 * 0^4 = 0$$
$$y_{B(0)} = (1-0)^4 * 0 + 4 * 0 * (1-0)^3 * 8 + 6 * 0^2 * (1-0)^2 * 8 + 4 * 3 * (1-0) * 0^3 + 0 * 0^4 = 0$$
$$P^I = 0; 0$$

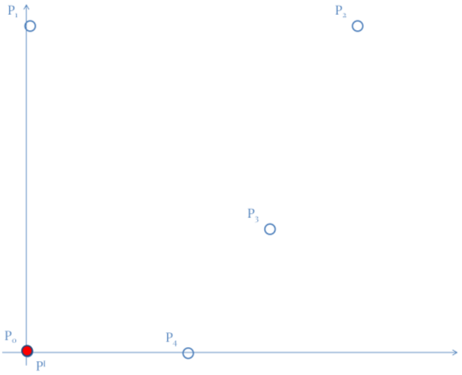


Figure B. Point P^{I}

$$x_{B(0.25)} = (1-0.25)^4 * 0 + 4 * 0.25 * (1-0.25)^3 * 0 + 6 * 0.25^2 * (1-0.25)^2 * 8 + 4 * 6 * (1-0.25) * 0.25^3 + 4 * 0.25^4 = 1.9844$$
$$y_{B(0.25)} = (1-0.25)^4 * 0 + 4 * 0.25 * (1-0.25)^3 * 8 + 6 * 0.25^2 * (1-0.25)^2 * 8 + 4 * 3 * (1-0.25) * 0.25^3 + 0 * 0.25^4 = 5.2031$$
$$P^{II} = 1.9844; 5.2031$$

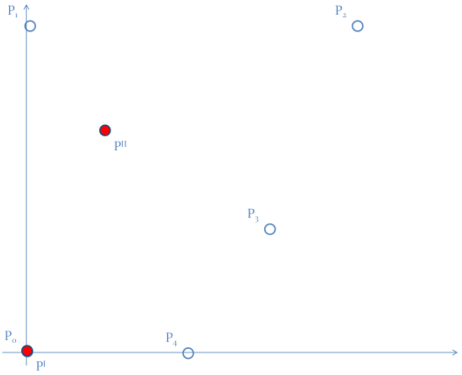


Figure B. Points P^{I} and P^{II}

$$x_{B(0.5)} = (1-0.5)^4 * 0 + 4 * 0.5 * (1-0.5)^3 * 0 + 6 * 0.5^2 * (1-0.5)^2 * 8 + 4 * 6 * (1-0.5) * 0.5^3 + 4 * 0.5^4 = 4.75$$
$$y_{B(0.5)} = (1-0.5)^4 * 0 + 4 * 0.5 * (1-0.5)^3 * 8 + 6 * 0.5^2 * (1-0.5)^2 * 8 + 4 * 3 * (1-0.5) * 0.5^3 + 0 * 0.5^4 = 5.75$$
$$P^{III} = 4.75; 5.75$$

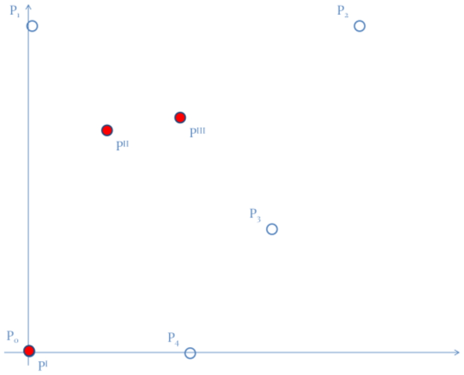


Figure B. Points P^{I}, P^{II} and P^{III}

$$x_{B(0.75)} = (1-0.75)^4 * 0 + 4 * 0.75 * (1-0.75)^3 * 0 + 6 * 0.75^2 * (1-0.75)^2 * 8 + 4 * 6 * (1-0.75) * 0.75^3 + 4 * 0.75^4 = 5.4844$$

$$y_{B(0.75)} = (1-0.75)^4 * 0 + 4 * 0.75 * (1-0.75)^3 * 8 + 6 * 0.75^2 * (1-0.75)^2 * 8 + 4 * 3 * (1-0.75) * 0.75^3 + 0 * 0.75^4 = 3.3282$$

$$P^{IV} = 5.4844; 3.3282$$

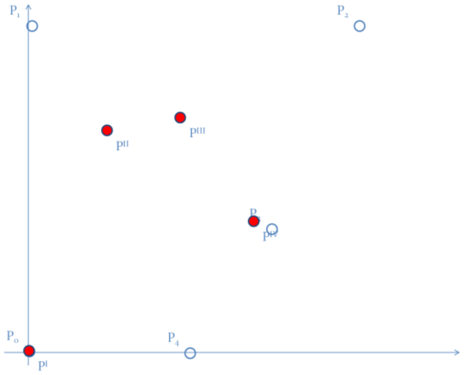


Figure B. Points P^{I}, P^{II}, P^{III} and P^{IV}

$$x_{B(1)} = (1-1)^4 * 0 + 4 * 1 * (1-1)^3 * 0 + 6 * 1^2 * (1-1)^2 * 8 + 4 * 6 * (1-1) * 1^3 + 4 * 1^4 = 4$$

$$y_{B(1)} = (1-1)^4 * 0 + 4 * 1 * (1-1)^3 * 8 + 6 * 1^2 * (1-1)^2 * 8 + 4 * 3 * (1-1) * 1^3 + 0 * 1^4 = 0$$

$$P^{V} = 4; 0$$

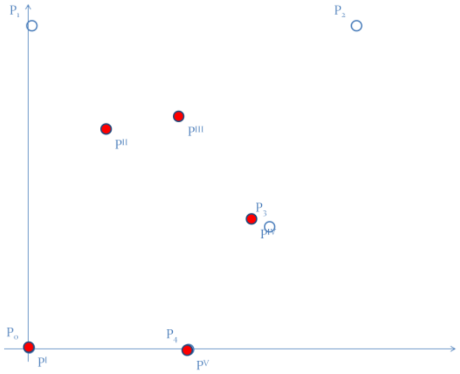


Figure B. Points P^{I}, P^{II}, P^{III}, P^{IV} and P^{V}

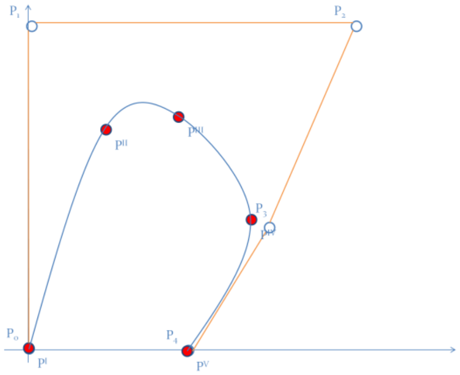


Figure B. Points P^{I}, P^{II}, P^{III}, P^{IV} and P^{V}, Bézier spline (in light blue) and straight line (orange)

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.

CHAPTER 10

Conclusion

The techniques used to obtain the 3D models are different. It is fundamental not only use the correct techniques, but also to apply them in the exact order.
In the following, a brief summary and the conclusions for each type of input point cloud. As mentioned many times in this dissertation, the goal is to obtain 3D models if the input point cloud represents a structure, an infrastructure, an aquifer or a shrub. These types of input point clouds were analyzed through the use of four study cases. The fist study case is the China Central Television Headquarter, the second is an international airport in Italy, the third is the aquifer of the city of Milan and a nearby area, the fourth one is a double study case. It compirises the data of Levy County and Marion County, Florida, U.S.A.
If the input point cloud represents a structure, the integrated techniques and the procedures used are the following:

1. check the input data set,
2. the “voxel” procedure, a method based on octree encoding, is used,
3. cluster analysis,
4. extract features from the data set,
5. relational matching,
6. Delaunay triangulation.

The software used is Fortran® and Matlab®.
The first procedure is to check the input data set. In this specific case there are no duplicate points.
The goal of the second procedure is to divide points into voxels and to estimate the values of three parameters. In this specific case, the optimal order of voxel is the 6° and has 1,802 points. The three parameters produced are the height interval equal to 7.969 m, the planimetrical distance equal to 17.678 m, and the planimetrical tolerance between points at different levels equal to 4.000 m.

The third procedure is devoted to analyzing the data in terms of cluster analysis. It is a two steps procedure. The first one is necessary to do vertical clustering, the second one is necessary to do horizontal clustering. In this case, there are 52 vertical clusters and 83 horizontal clusters.
The fourth procedure is devoted to distinguishing chains from sowns. In this case there are 4 sowns and 79 chains.
The goal of the fifth procedure is to analyze the data set in terms of relational matching. It is done separately for sown – chain and for chain – chain. In the reconstruction of CCTV, the chain sides matched are 148, the sown sides matched are 18 and the total number is 166.
The sixth procedure is devoted to building closed loops between all chains and all their projections on sowns. A switch parameter should be used in order to choose which type of interpolation. In my opinion, in this specific case the closed loops are modeled using the classic way. If the classic perimetration is set, the connection between the points doesn't generate other new points.
The seventh procedure is devoted to connecting points into triangles. The triangulation must connect all the points belonging to a single horizontal cluster. If one horizontal cluster has a hole, the triangulation must have the same hole. If one horizontal cluster is concave, the triangulation cannot contain extra sides.

If the input point cloud represents an infrastructure, the integrated techniques and the procedures used are the following:

1. check the data set and to provide the first 3D plots,
2. generate the 3D model,
3. analyze the 3D model and the point cloud in terms of clusters,
4. relational matching, or connecting the points that belong to different clusters,
5. connect the points with a Delaunay triangulation,
6. Brückner Profiling, or evaluating the volume of soil to be handled.

The aim is to detect and remove the outliers and to identify and to isolate the center line. It is also possible to estimate the value of space parameter using the total number of points and the fact that the points are equally spaced along the area. In this case the space parameter is equal to 10 m.
An airport is composed of several parts. The widths and the cross slpoes of each element depend on the Aerodrome Reference Code. In this particular case, the Aerodrome Reference Code is 4E. The aim is to obtain a planimetric model with the required widths and then a 3D model with the required cross slopes.
The third procedure is devoted to analyzing the data in terms of clusters. The cluster analysis procedure is done separately for the 3D model and for the input data set.
Because of the density of the point cloud, it is impossible to detect some parts of the airport.For this reason, it is necessary to interpolate the data.
At this point, the data set is analyzed in terms of relational matching. In this specific case, a perfect mapping between each point is attempted so the matching type is a one to one.
Several relational matching iterations are completed. The first one involves all the points of the point cloud and it is done in horizontally. The second relational matching involves all the points of the 3D model and it is done horizontally. The third type of relational matching is done vertically.

Points with the same y-coordinate that belong to the point cloud are matched with the same y-coordinate that belong to the 3D model.

The last three types of relational matching can be combined to form a set of four points. Two points belong to the 3D model, two points to the point cloud. The four of them are connected using relational matching. The fourth type of relational matching is done vertically, considering the cluster analysis previously done. Points that belong to the point cloud and are classified as belonging to the same cluster are matched with the points that belong to the 3D model and are classified as belonging to the same cluster.

Without these types of relational matching it would be impossible to obtain the area diagrams automatically.

The fifth procedure is devoted to connecting the data with Delaunay triangulation. It is necessary to apply Delaunay triangulation in two different ways. The first way is applied horizontally to the point cloud and to the 3D model. The second method is applied to sets of four points. Those sets are generated with the first three types of relational matching earlier described.

the sixth procedure is devoted to obtaining the area diagrams. To estimate the value of each section, the sum of the areas of two triangles is required. The values are obtained using Erone formula. All the areas are positives. It implies there are no fills; the airport will be in cut sections.

The seventh procedure is devoted to optimizing the volume of the soil to be handled. To do this, the Brückner profile tool is used.

Because all the areas are positive, all the volumes are positives. It is not possible to optimize the movements of the soil from the cut sections to the fill sections.

If the input point cloud represents an aquifer, some problems must be solved to obtain a consistent 3D model. The first problem originates from the fact that if the input data contains some outliers, the predicted values will be wrong. The second problem relates to the fact that the input data are discrete, not equally spaced, uniform or homogeneous and the outputted model must be continuous. This fact implies that the model is obtained using interpolated techniques.

It is possible to solve these problems subdividing the data processing into six steps.

First of all, the input data are checked in order to detect and remove the outliers.

Secondly, the information is divided into clusters.

Thirdly, each cluster is interpolated using Delaunay triangulation.

Fourthly, closed paths between points that belong to a specific layer are built. It is necessary to bounds each layer in order to know where to interpolate and where to extrapolate. The method adopted is the following: the Delaunay triangulation generates triangles between points. Each triangle is composed of three sides. If a side is in common with more than one triangle, it is deleted. On the contrary, the side is classified as a side of the closed path.

Fifthly, it is necessary to search the correct correspondences between points that belong to consecutive layers to generate the vertical boundary. This procedure is carried on using the relational matching method.

Sixthly, it is necessary to obtain new information starting from the Delaunay triangulation. This procedure is carried on using the interpolation method.

If the input point cloud represents a concave, multi-connected and non-stellar body, it is necessary to apply first the feature extraction procedure and then Delaunay triangulation method. If the input point cloud represents this aquifer, it is necessary to apply Delaunay triangulation procedure and then the feature extractions method.

If the input point cloud represents a shrub, the input point cloud analyzed is a raw point cloud data in LAS 1.1 format. The raw point cloud LiDAR is divided into 733 tiles. The input data contain also breaklines.

In order to sub-classify class 1, different tools in this specific order have to be used:

1. LAS to multipoints: this tool imports all the files in LAS format into a new multipoint feature class. File formats supported are LAS 1.0, LAS 1.1 and LAS 1.2. The input data is in LAS 1.1 format. The LAS to multipoints step is applied twice: the first time to obtain the non-ground points using the Model Builder "LAStoMPng". The second time to obtain the ground points using the Model Builder "LAStoMPg".
2. Point to raster: the point to raster converts point features to a raster dataset. The point to raster step is applied twice: the first time to obtain the Digital Surface Model (DSM), the second time to obtain the Digital Elevation Model (DEM). The Point to raster step is applied using the Model Builder "NGtoDSM" and "GtoDEM".
3. This step is used to obtain the relative height of all the points. The a python script "nDSM" is used.
4. Raster to point: the raster to point conversion tool converts a raster dataset to point features. The Raster to Point Conversion tool converts a raster dataset to point features. For each cell of the input raster dataset, a point will be created in the output feature class. The points will be positioned at the centers of cells that they represent. The NoData cells will not be transformed into points. To convert the raster into points, the python "RasterToPoint" script is used.
5. Select by attributes: it adds, updates, or removes a selection on a layer or table view based on an attribute query. To select all the points within a certain height, Select Layer By Attribute (Data Management) is used. This step is applied four times:
    a. vegetation high lower than 0 ft
    b. Height between 0 ft and 3.28 ft
    c. Height between 3.28 ft and 6.56 ft
    d. Height between 6.56 ft and 19.658 ft

   These steps are carried on using the "LowerThanZero", "LowerThanThree", "LowerThanSix" and "LowerThanNinteen" python scripts.
6. Split: splitting the input features, it creates a subset of multiple output feature classes.
7. Feature to polygon: it creates a feature class containing polygons generated from areas enclosed by input line or polygon features.
8. Clip: it extracts input features that overlay the clip features. This tool cuts out a piece of the roads feature class using the footprint feature in the new feature class called clipped roads.
9. Selects features in a layer based on a spatial relationship to features in another layer.
10. Point density: it calculates a magnitude per unit area from point features that fall within a neighborhood around each cell.

References

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

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

http://gisnetwork.provincia.ms.it/CARTOTECA/Sistemidiproiezione.aspx

Anderson, H.E., 1982. Aids to Determining Fuel Models for Estimating Fire Behavior. U.S. Department of Agriculture Forest Service Research Note, INT-122, National Wildfire Coordinating Group, 22 p.

Stanley Coren, 2005. How Dogs think. What the world looks like to them and why they act the way they do. Free Press, New York. NY 10020

https://www.e-education.psu.edu/LiDAR/book/export/html/1873

http://www.alzheimer.ca/docs/brochure-search-is-an-emerg-eng.pdf

http://www.dawgs.org/faq.html

http://www.catherinedold.com/portfolio/feature-articles/rescue-dogs/

http://familyarticles.info/parenting/gps-device-and-how-it-helps-rescuers-save-missing-people

http://www.mra.org/

Agee et al., The use of shaded fuelbreaks in landscape fire management

PE&RS, March 2011.Volume 77, number 3.

http://nsidc.org/data/icesat/tools.htmlPatenaude, G., Hill, R. A., Milne, R., Gaveau, D. L. A., Briggs, B. B. J., & Dawson,

http://www.isprs.org/proceedings/XXXVI/part7/PDF/185.pdf

http://www.sciencedirect.com/science/article/pii/S0924271605001024

http://www.firemodels.org/index.php/national-systems/farsite

http://www.csiro.au/en/Outcomes/Safeguarding-Australia/Infrastructure-Bushfires.aspx

http://www.imagingnotes.com/go/article_freeJ.php?mp_id=264 http://ceal.warnercnr.colostate.edu/LiDAR_Background.htm

http://terraformatics.blogspot.com/2009/11/full-waveform-als-workshop-radiometric.html

http://www.forestencyclopedia.net

http://www.idav.ucdavis.edu/education/CAGDNotes/Bernstein-Polynomials/Bernstein-Polynomials.html

http://en.wikipedia.org/wiki/B%C3%A9zier_curve

T. Bonomi, P. Canepa, F. Del Rosso. Stima dei volumi di acqua in un acquifero eterogeneo: l’esempio della provincia di Milano. Rendiconti online Società Geologica Italiana. Vol. 8 (2009), 19 – 22.

N.A. Cressie. Statistics for spatial data. John Wiley and Sons, New York, 1991.

P. Gattinoni. Analisi della struttura spaziale di variabili idrogeologiche con tecniche geostatistiche. Quaderni di geologia applicata, 2004.

G. Naldi, L. Pareschi. Matlab, concetti e progetti. Apogeo, 2007.

R. Pratap. Matlab 7. A quick introduction for Scientists and Engineers. Oxford University press, 2006.

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

http://home.dei.polimi.it/matteucc/Clustering/tutorial_html

http://www-users.cs.umn.edu/~kumar/dmbook/ch8.pdf

SHRUB

Anderson, H.E., 1982. Aids to Determining Fuel Models for Estimating Fire Behavior. U.S. Department of Agriculture Forest Service Research Note, INT-122, National Wildfire Coordinating Group, 22 p.

Stanley Coren, 2005. How Dogs think. What the world looks like to them and why they act the way they do. Free Press, New York. NY 10020

https://www.e-education.psu.edu/LiDAR/book/export/html/1873 http://www.alzheimer.ca/docs/brochure-search-is-an-emerg-eng.pdf

http://www.dawgs.org/faq.html

http://www.catherinedold.com/portfolio/feature-articles/rescue-dogs/

http://familyarticles.info/parenting/gps-device-and-how-it-helps-rescuers-save-missing-people

http://www.mra.org/

Agee et al., The use of shaded fuelbreaks in landscape fire management

PE&RS, March 2011.Volume 77, number 3.

http://nsidc.org/data/icesat/tools.htmlPatenaude, G., Hill, R. A., Milne, R., Gaveau, D. L. A., Briggs, B. B. J., & Dawson,

http://www.isprs.org/proceedings/XXXVI/part7/PDF/185.pdf

http://www.sciencedirect.com/science/article/pii/S0924271605001024

http://www.firemodels.org/index.php/national-systems/farsite

http://www.csiro.au/en/Outcomes/Safeguarding-Australia/Infrastructure-Bushfires.aspx

http://www.imagingnotes.com/go/article_freeJ.php?mp_id=264

http://ceal.warnercnr.colostate.edu/LiDAR_Background.htm

http://terraformatics.blogspot.com/2009/11/full-waveform-als-workshop-radiometric.html

http://www.forestencyclopedia.net