AUTOMATED BUILDING DETECTION IN DENSE POINT CLOUD AND UPDATE OF OPEN SOURCE DATA BASES

E. Aroni, C. Ioannidis

Laboratory of Photogrammetry, School of Rural and Surveying Engineering, National Technical University of Athens, Greece
eriaaroni11@gmail.com, cioannid@survey.ntua.gr

KEY WORDS: Dense matching, Building detection, Classification, Digital Terrain Model, openstreetmap, point cloud

ABSTRACT:

In this paper a method of detecting buildings in dense populated city areas using a three-dimensional model, produced by aerial images, is described. Further to the detection of the outline of the building, we exact information about the buildings height. The study area is the wider centre of Athens, Greece. Our aim is to exact 3D information for large area, in minimum time and minimum cost, in order to support opensource data bases, such as openstreetmap.org. The proposed methodology consists of three main stages. In the first part of the procedure, aerial images are used to produce a point cloud, using the Semi-Global dense matching algorithm. Following, we classify the objects in the point cloud by remote sensing and photogrammetric methods. The classification's results are divided in three main classes: ground, vegetation and buildings. Having detected the buildings and their complexes we attempt to find the outlines of each separate building, depending on its level; different levels are considered as different buildings. After detecting individual buildings in the point cloud, a polygon is created around their outline. All polygons were compared to the building polygons available on openstreetmap.org, in order to evaluate the results. The number of levels of 100 buildings, in different parts of the city, was measured manually in order to evaluate the Z-dimension's results, and openstreetmap.org was updated with that information. Further update and combination of the database created in the current process, with the one available on openstreetmap.org is yet under study.

1. INTRODUCTION

Building detection and description is a task with various applications in the fields of GIS, topography, cartography and urban planning. More often, buildings are mapped using polygons describing their outline. Depending on the initial data and the procedure, different approaches have been taken in automatically mapping buildings, using aerial or satellite images. Muller and Zaum (2005) propose a single-image building detection, based on RGB aerial images. During the pre-processing, the image is transformed to HIS, and following, using image segmentation all regions are detected. In the post processing step regions fulfilling special conditions are merged.
In a different approach, Liu and Prinet (2005) use a probability model in satellite images. They detect the buildings assuming their space distribution is represented by a logistic function. Each building is detected as an independent object in a single image, based on its similarity to proposed features. Using also single-image data, Nevatia and Ramakant, (1998) detect building polygons based on the linear edges of their rooftops. Combining the verified results with information taken from the buildings' shadows in the image, they reach conclusions concerning all three dimensions of the building. More recently, edge detection and Haar features were used in developing a scale and rotation invariant method that detects building in low quality images (Cohen et al., 2016).

Other initial data used often are LIDAR- sensor data, by creating dense point clouds and detecting planes (Rottensteiner, 2003; Rottensteiner et al., 2007). Point clouds from LIDAR sensors, in contrast with dense cloud created by photogrammetric methods, contain fewer outliers, forming more easily detected geometrical shapes. Remote sensing methods have been applied in an attempt of classifying objects in urban areas, both in satellite (Zhang, 1999) and aerial images, combined with spatial data. The combination is required due to the non- uniformity of buildings usually existing in urban areas. Satellite images containing multispectral information, combined with deep learning algorithms have been used to determine a binary classification method- "building" or "not building" (Vakalopoulou et al., 2015). Ground truth datasets were initially used to train the algorithm, which was then applied on modified and corrected satellite images.

Detecting a single building in an image is a very complexity task (Kiran, 2015). Depending on the city's architecture, each building could consist of a single parallelepiped and its roof in the simplest case, or, in the most complicated version of the problem, many buildings create a complex, having no spaces between them. Moreover, a single building might have more than one different levels (e.g., penthouses) or objects on the rooftop that affect both its outline and its estimated height. Finally, a very common issue, especially in Mediterranean architecture, is the balconies that can be detected as parts of the building in many different levels, altering the results of the description.

Athens is a city where all of the above forms of buildings can be found within a few meters from each other. In areas around main highways, buildings are high and bulky, having terraces and fewer balconies. On the other hand, in older parts of the city centre, buildings are up to two floors, they usually have rooftops, balconies and many different parts. Furthermore, in most parts of the city's centre, buildings are located right next to each other, making it even more difficult to demarcate them. In the current project a combination of remote-sensing classification methods, along with geometrical characteristics and GIS techniques are used in order to locate individual buildings, map their outline and calculate the number of levels each building consists of.

2. PROPOSED METHODOLOGY

The methodology developed in this project aims in detecting buildings in dense populated areas of large cities. The initial data required for the process are at least two overlapping aerial images with known interior and exterior orientation. Images with known relative orientation require extra information of three ground control points visible in both images. In Figure 1 a flowchart is presented, depicting the main stages of the procedure, which are further described in the following sections.

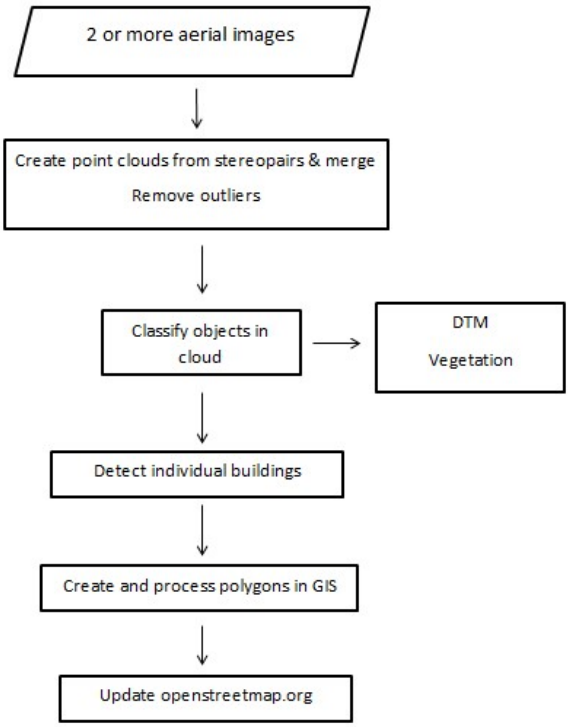

Figure 1: Main stages of the process

2.1 Dense matching

The minimum data required for the process is a stereopair of images. More than two overlapping images can provide even better results, decreasing non-visible areas in the point cloud, as long as the geometry of the block is appropriate. The overlap between the images has to be arranged depending on the flight height and the focal length. Each image used in the current process contained information about the radiance value in four bands: red, green blue and infrared band. The infrared band is of great importance in the stages of classifying the objects and detecting vegetation. Exterior and interior orientation were calculated for all images.

By applying the semi-global dense matching algorithm (Haala, 2013; Ernst and Hirschmüller, 2008; Dall'Asta and Roncella, 2014), combined with the above information, a dense point cloud is created for each separate stereopair. All four bands' values are stored in the created point cloud, along with the position for each point's position (X, Y, Z) in a fixed reference system. The semi global algorithm is applied on two images, therefore in large areas covered by more than one stereopairs, multiple overlaying point clouds are created.

Following, the overlaying point clouds are merged into a single cloud. A downsampling method is applied at the overlapping areas, in order to eliminate redundant points. In occasion, the same ground point is visible in more than two images, and, therefore, in more than one point cloud. Duplicate recordings lead to overloading datasets and represent the same ground point in slightly different colours and positions in each cloud. As a result, it was a necessity to calculate a unique value for the position and the radiance of the points. Using a downsampling function, the average position and radiance value for each point is calculated within a certain area (grid box). Considering the ground pixel size of the cloud, the grid box is selected at a slightly smaller size, to maintain the initial spatial resolution of the cloud. The current application aims on extracting information using the minimum possible data required. In cases where multiple image coverage is possible, the redundancy of images and, therefore, points can be used to obtain further information and instead of Semi-global techniques other algorithms may be applied, such as Structure from Motion technique.

The results of the semi global algorithm are affected by a number of factors during the process. Commonly, due to the geometry of the central projection, certain ground points might be visible in only one of the two images of the stereopair. Furthermore, correspondences in the two images are not always well estimated. Consequently, the position and the colour of some points in the point cloud do not match their true value, as expected. When visualizing the point cloud, these outliers are detected, since they appear to be far away from the main landscape. Removing these outliers requires the calculation of the clustering of the points into space (Vosselman and Dijkman, 2001). By counting the number of points within a certain sphere around each point, the algorithm determines whether a point is an outlier or an inlier. The minimum required number of points differs in each application, depending on the density of the cloud, and so does the range of search. The thresholds chosen may affect the result; strict limits lead to very slow performance and to elimination of points that are actually inliers, whereas high tolerance does not remove efficiently the outliers. By comparing the number of points in the point cloud before and after the removal, we determine a percentage of noise (outliers in the cloud) that was detected and removed. Outliers still will exist in the point cloud, and since their groups consists of a larger number than the threshold chosen, the remaining percentage of noise will possibly be even higher than the removed. This information will be used in the next stages of the procedure.

2.2 Classification

The second part of the procedure aims in detecting the complexes of the buildings in the point cloud. Remote sensing methods are particularly effective in detecting objects such as vegetation, water and ground, especially when infrared data are available. In cases of urban areas, though, the spectral responses of different objects may be quite similar. More specifically, ground is mostly covered with constructions such as pavements, asphalt roads etc. As a result, the colour values are usually low in infrared and medium in all other bands, just like concrete and buildings. Except for the vegetation, therefore, all the other classes in the cloud are difficult to detect. In order to classify all the objects, both remote sensing methods and geometric properties of the objects are used.

The main idea of this classification is to detect all the buildings by excluding from the cloud all the other categories. The objects were divided in four major classes: ground, vegetation,

buildings with terraces and buildings with roofs. The classes were created based on the different spectral responses of the objects or their different place and shape in the cloud. The "ground" class contains all the roads and the constructions with no elevation from the ground, along with the bare ground surface. The "vegetation" class includes all the plants and trees. Finally, buildings are divided in two sub-classes, those with roofs and those with terraces, due to their different shape and colour.

The first step is to detect the ground surface and to create a digital terrain model (DTM) of the city. Due to the similarity in reflectance with the buildings, in order to model the ground surface without using additional data, an algorithm was developed based on the geometry of the ground. Precisely, the idea was that when creating a point cloud from aerial images, ground is the surface with the minimum altitude in the cloud (in lack of transparent objects). Based on that idea, in certain areas of the cloud we search out for the altitude of the lowest level. There are, though, two potential exceptions:

- At least part of the ground has to be visible in the search area; otherwise the lowest level could belong to a building or vegetation.
- The remaining noise in the cloud can set the minimum altitude of an area much lower than its true value.

In order to overcome these exceptions, the algorithm firstly divides the entire cloud in parts large enough to contain at least a ground surface (e.g., roads). By measuring the side of one large block in the cloud, we set the one side of the above parts slightly larger, in order to contain at least one part of roads or ground. The other side of the area is set at half the size to achieve a denser sampling. The segmentation is applied parallel to both X and Y direction. Consequently, the created grid density is half the size of the measured block. In each of the segmented areas, instead of the lowest altitude, the algorithm starts searching bottom to top for the lowest level that does not consist of remaining outliers. The condition that ensures this hypothesis is that the sum of the points found is larger than the previously estimated remaining noise level. Accordingly, any object consisted of more than that percentage is less likely to be noise. A slightly higher threshold than the estimation provides a more solid result. Starting from the lowest Z in the area, it adds up all the points between that minimum Z and a distance provided by the user. If the number of the points is less than the threshold the algorithm proceeds to a higher level, as shown in Figure 2.

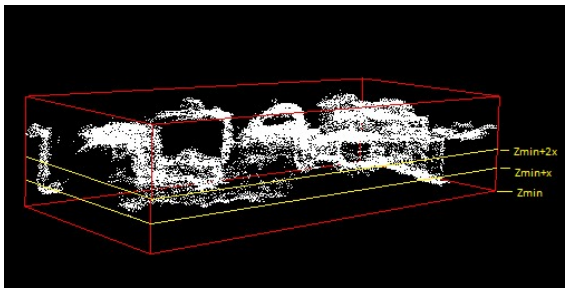

Figure 2: The algorithm searches, starting from the Zmin and for two more iterations, for the lowest level

The distance between each search defines the accuracy of the estimation of the surface's position. At the same time, though, the smaller the step, the more iterations take place, and the slower the performance is. The user is able to define the search step depending on the project's requirements. When the minimum level is found all points with an altitude of 1.50 meters difference from that level are classified as "ground points" and are saved in a different file. Following the detection of the ground points, the digital terrain model is created by triangulating the new point cloud. Smoothing the produced surface is recommended (Figure 3). For each other point of the initial cloud, its distance from the DTM is calculated as a scale value (Balenović et al., 2016).

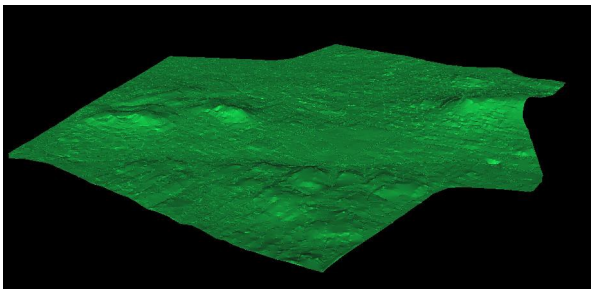

Figure 3: DTM after triangulation and smoothing

Objects with low elevation from the DTM are removed from the cloud. In urban areas several constructions may have been recorded in the images and the cloud, such as street lamps, kiosks, signs etc. The more detailed the image, the more the objects. Since there is no uniformity in these constructions, they can not be classified. By deleting all points between a certain distance from the ground, all these objects are eliminated from the next steps of the procedure. Taking into consideration each city's architecture, a threshold in elevation is determined so that minimum loss of useful information (actual buildings) is experienced. Points with extremely high (absolute) elevation values are also deleted, since they are more likely remaining outliers.

In the second part of the classification we calculate the Normalized Difference Vegetation Index for each one of the points in order to detect and eliminate the vegetation. The index is described by the Equation 1.

$$N = \frac{f(i,4)-f(i,3)}{f(i,4)+f(i,3)} \qquad (1)$$

f(i,3) is the value for the red band and
f(i,4) the infrared band's value.

Vegetation is recorded with high values in the infrared band, with a suddenly reduced record in the next band, the red. The normalized ratio between the two bands was used to eliminate the vegetation. High values of the NDVI are recorded both for vegetation and buildings with roofs. By sampling in the cloud's values, a minimum threshold is determined for "vegetation" class. Usually, a very strict threshold eliminates the major part of vegetation along with the roofs, due to both having high NDVI values. A second criterion, therefore, of distinguishing those buildings is combined with the NDVI in order to avoid data loss: roofs appear in red for the RGB 3,2,1 palette, whilst vegetation appears in green. A ratio between bands 2 and 3 should be higher than 1 for one class and lower than 1 for the other. Either the ratio 2/3 or the reverse, one of the classes is recorded with higher than one (>1) results. In cases where water is visible in the cloud (rivers, coasts etc.) a second (minimum) threshold is required in the NDVI, since water is recorded with significantly low index values.

2.3 Creating polygons

Having completed the classification of objects in the cloud, the last stage is to create the polygons around each building and create a GIS database for all the buildings. Such a result required that each building is located separately from the rest of the buildings. After locating unique buildings in the point cloud, their boundaries are enclosed with a polygon, while storing their scale-information (elevation from the ground) calculated in the previous step.

A modification of the algorithm that was created to locate the lowest level (DTM) is used to separate the buildings within their complexes. Buildings with roofs are found in less densely populated areas, and in most cases already have enough spaces between them. Their polygons are, therefore, created just by locating the clusters on the XY plane after the classification and enclosing them with a boundary (Figure 4). The attributes contain three-dimensional information. Instead of the absolute Z information, in the data type (PolygonZ in the current project) the elevation from the ground (height of the building) is stored. In cases where there are no spaces between the buildings, RANSAC algorithm can be used to locate the planes of the roofs (Tarsha-Kurdi et al., 2008)

Figure 4: Individual buildings with roofs

On the contrary, buildings with terraces are usually in complexes, therefore clustering on the XY plane leads to polygons describing more than one building (Figure 5).

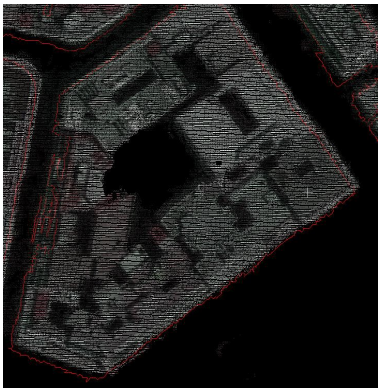

Figure 5: Complexes of buildings with terraces

Describing unique buildings within the complexes requires a criterion of separation. Computers are unable of recognizing whether one building has different parts with various colours or levels, or each of these parts is a separate building. Since the colour- value is similar for all the buildings, the criterion chosen was the level. Structures of different levels are considered to be different buildings and are described by different PolygonZ attributes.

In order to locate the level where each terrace was, in the classified cloud, a modification of the DTM algorithm was created. Firstly, the entire cloud was segmented in parts- the size of one or two blocks offers best combination of time and results. In each of the parts, the algorithm searches for all the possible levels, and stores each building in a different file. Instead of searching for the minimum level, though, the equation was reversed and the algorithm started top to bottom adding the points from the maximum Z. The iteration step was once more provided by the user and the iterations were terminated as soon as the highest level was detected (Figure 5-top). The initial required percentage is reduced to half; since the amount of noise is lower in the classified cloud (points with large elevation have been deleted). Each time the algorithm detects enough points to be considered as a new level and not noise, it saves that level in a separate cloud and removes it from the total cloud (Figure 5-bottom). The process continues for the rest of the points, and in each iteration i the required percentage is increased by i/2. The increase is necessary, because by reducing the total number of points in each loop, every remaining level occupies a larger percentage in the entire sub-cloud. The process is terminated when the level found is equal to the minimum Z of the segmented area. Each of the buildings is described by a polygon, as in the buildings with roofs. In the attribute table of the polygons, apart from the elevation from the ground, the number of levels is calculated, by dividing the calculated elevation by an estimated floor's height, and rounding it to the closest integer. The floor's height varies, depending on the city's urban planning design.

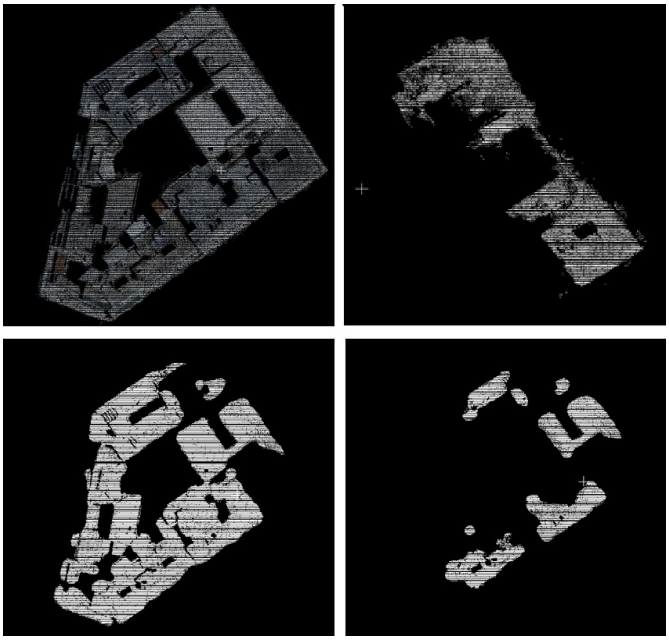

Figure 6: Total cloud (top left), first highest level (top right), rest cloud (bottom left) and second highest level (bottom right)

2.4 GIS processing

Post- processing of the polygons in a GIS is a necessity due to various factors. First of all, by segmenting the cloud into smaller sub-clouds some buildings are cut in two or more pieces, each one in a different subset. The first required

correction, consequently, is to regenerate those buildings (Figure 7). Polygons of the same level overlapping or closer than 0.5 meters are dissolved into a single polygon in order to rectify the original building.

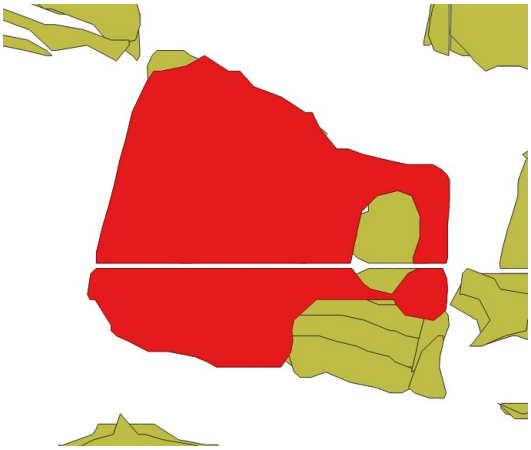

Figure 7: Segmented building

The second required processing is to eliminate parts of the buildings with flawed geometries, such as balconies, parts of noise between different structures etc. Those elongated polygons are rectified by converting all the polygons to polylines. The polylines are buffed to a certain distance to the inner side and then re-buffed to the outside of the polygon, as described in Figure 8. The blue line describes the original polygon and the black dashed line is the inner buff. The red line describes the rectified polygon. The size of the buff is determined based on the sizes of the buildings in the area.

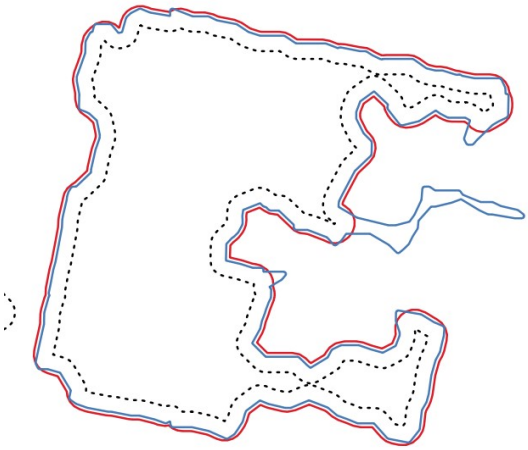

Figure 8: Eliminating elongated polygons

As a result, parts of polygons with at least one side shorter than the selected buff size are deleted from the final outcome.

Finally, due to the fact that polygons are extracted from a three-dimensional point cloud, apart from the buildings terrace or roof, parts of their front view are visible in lower levels and, therefore, enclosed in polygons. Those polygons overlap with the roofs and the higher levels, leading to more than one level-information for the same XY position (Figure 9).

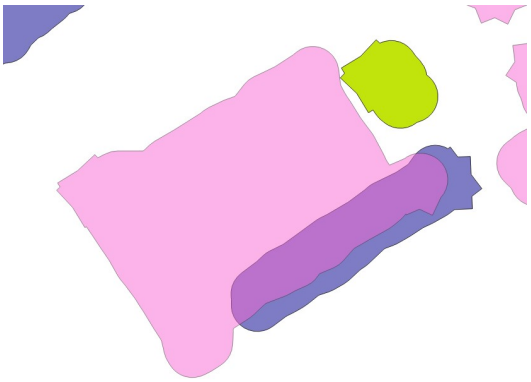

Figure 9: Overlapping areas of different level values

Regarding all those overlapping areas any information below the highest level is deleted and stored only for the polygon of highest floor.

3. IMPLEMENTATION

The study area was in the center of Athens, Greece, covering an area of around 10 km^2. The area was partially covered with thick vegetation and archeological sites. Moreover, the terrain was intense in the northern parts and smooth in the south- west.

The initial data in the current project were eight (8) overlapping aerial images. The images were taken in two different flight strips, oriented vertically to each other. The exterior orientation of the images was known beforehand, and each image had information in four bands (red, green, blue and infrared band).

3.1 Dense matching

In the first stage of the implementation semi-global algorithm was run using the software Erdas Imagine. Stereopairs were created by images of the same strip in order to avoid interdictions with inaccurate geometry. Totally, six stereopairs, and therefore, six different point clouds were created. The radiance value and the absolute position (X,Y,Z) of the points were stored in each point cloud. The reference system used is GGRS ’87 and the estimated precision of the coordinates is 0.04 meters.

Having created the individual point clouds, they were all merged by an algorithm created in Matlab. Following the merging of the clouds, downsampling was applied by calculating the average of color and position within a box. Considering the ground pixel size of the cloud, which was 0.25 meters, the grid box was selected at 0.24 meters, to maintain the initial spatial resolution.

Outlier removal was also run using a Matlab function. In this particular application, the threshold was 45 points within one meter from each point. The threshold was determined experimentally after testing the algorithm both with larger and smaller thresholds. Comparing the total number of points before and after the outliers’ removal, the noise was estimated around 2% of the cloud’s points. Parts of the final outcome of the procedure’s first part are presented in Figure 10, in RGB 4,3,2.

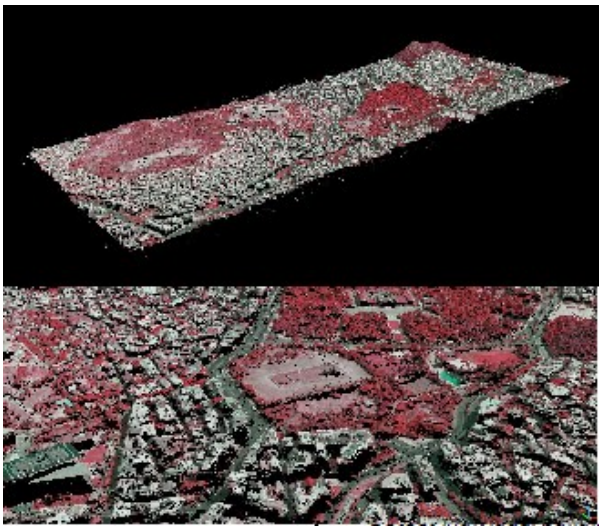

Figure 10: Parts of the created point cloud

3.2 Classification

During the first part of the classification a new point cloud was created by applying the algorithm described in chapter 2.2. The larger side of a block in the entire study area was measured at 70 meters; therefore, an area at least 100 meters long should contain one or more roads. The other side of the area was set at half the size (50 meters) to achieve the required denser sampling (Figure 11).

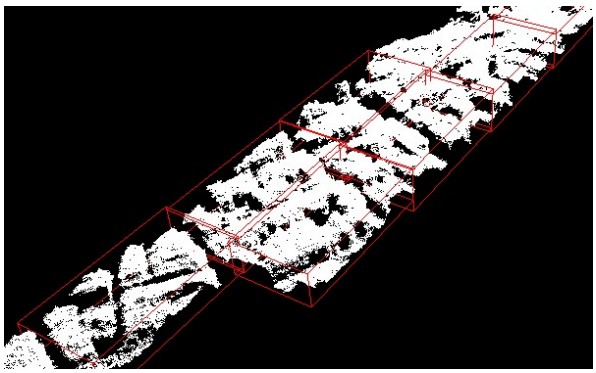

Figure 11: Segmenting part of the cloud in sub-clouds

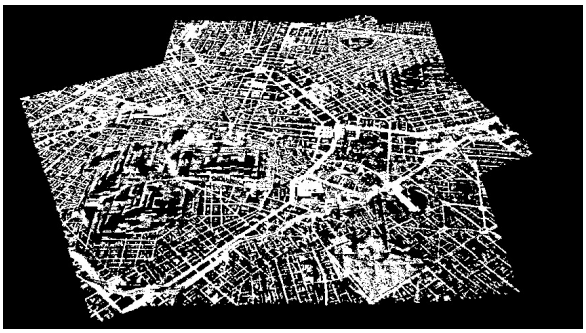

Figure 12: Point cloud including ground-only points

As described, the noise level was estimated around 2% during the outlier removal. Accordingly, any object consisted of more than 2% of the sub-cloud's total points was less likely to be noise. A threshold of 5%, for a more solid result, was determined, and no color information was stored for this part of the algorithm. The results, after combining the outcomes in X and Y directions, for the whole study area are presented in Figure 12.

By triangulating and smoothing the surface, a Digital Terrain Model was created, the distance from which was calculated for every point in the initial cloud. Points distant from the ground less than 7 meters, belong to either one-level buildings or to unneeded for the project constructions, such as kiosks, traffic lights, sculptures etc. One- level buildings are rarely found in a centre of a city like Athens, whereas all the other constructions are very common and need to be eliminated from the final outcome. In this part of the process, these points were deleted from the cloud, with the remaining classes being: tall vegetation, buildings with terrace and buildings with roofs. Moreover, points with elevation more than 45 meters from the DTM were deleted as well, reducing significantly the amount of remaining noise in the cloud.

In the second part of the classification, NDVI was calculated for every remaining point. Figure 13 describes the values of each remaining class in the cropped cloud.

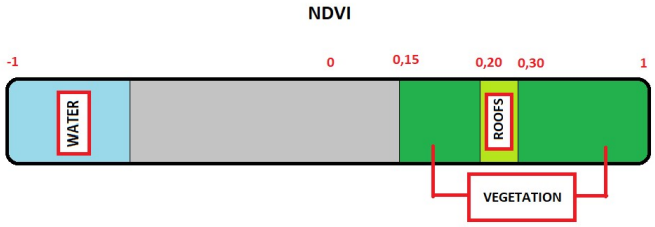

Figure 13: Several classes in the NDVI's scale

The diagram was created after sampling; high values of the NDVI were recorded both for vegetation and buildings with roofs. Using a threshold for all the points with an index lower than 0.15, all the buildings with terraces are separated from vegetation. Along with the vegetation, though, all buildings with roofs were eliminated from the cloud due to having high NDVI values. The ratio 3/2 was calculated as well, and was, as expected higher than (1) in the "roofs" class. By applying both indexes simultaneously roofs and vegetation were excluded from the cloud and saved in separate files, leaving the last remaining class, buildings with terraces in the final cloud.

The cloud including buildings with roofs was stored for further elaboration in the next stage, while for the cloud containing buildings with terraces further processing was required: the algorithm detecting the first level was applied to locate individual buildings within the complexes. Since the point cloud noise was reduced after the cropping, the threshold for accepting a level as large enough not to be an outlier was set at 2%. The algorithm was searching for levels every 0.30 meters, and each level was stored in a separate file. All algorithms were programmed and run in Matlab. Visualization, triangulation, smoothing and cropping of the point cloud were completed using CloudCompare.exe.

3.3 Polygon processing and GIS

Each separate building was stored in a different file, and buildings with roofs were stored all in a different cloud. Using LasTools (lasboundary.exe), a polygon was created around each clustered part of the above clouds. All polygons were stored in a merged shapefile and were processed in ArcGIS and QGIS. ArcGIS was used to extract the third dimension's information from the "PolygonZ" type features that were created in

LasTools, and store it in the attribute table. When locating the individual buildings instead of the actual Z position of the points, the third dimension was replaced with the distance from the DTM. The attribute table was updated with the distance from the ground, which separated by (3) states the number of different floors the building consists of (three meters is the average height of the floor according to Athens' town planning regulation). All polygons were inserted in QGIS, where the processes described in chapter 2.4 were applied. Firstly, overlapping areas were attributed to the highest level. Secondly, using a buff of three meters, elongated polygons were removed from the data base, along with parts of larger polygons with flawed geometry. Segmented buildings were regenerated, and each level was stored in a separate shapefile. All the polygons describing the detected buildings were evaluated using the existing polygons available on openstreetmap.org.

3.4 Openstreetmap update

The final aim of creating the database is to automatically update open source data bases, as openstreetmap.org. Polygons describing building on openstreetmap are created voluntarily by users, and in their majority describe sufficiently the buildings. Recently, fields describing the third dimension (height and level for each building) were added for each attribute. Combining the two databases is a task still in process, due to a number of difficulties.

First of all, polygons created in the current process are not identical to those on openstreetmap for various reasons. Users often digitize the buildings by describing the entire area around it, creating a polygon for the whole property and not just the building. Furthermore, users enclose in a polygon not only the main part of the building, but parts as balconies, garages and other collateral constructions. Moreover, as predefined, multiple polygons, created in this process, with different level-information, might belong to the same building, since the algorithm describes the buildings more thoroughly than the openstreetmap polygons. Finally, more than one buildings of the same level, having no space between them, will be enclosed in the same polygon by this algorithm, whereas in manual digitization will be described by two or more separate attributes. All of the above are the reasons that updating data sources with automated methods is a complexity issue that requires several assumptions.

One suggestion is to create a union of the two databases, and match each polygon to the highest elevation of the overlapping areas. For example, if one "openstreetmap" polygon overlaps with two different polygons created by the algorithm, we assign to the first the highest elevation value of the two. On the other hand, that highest-value overlapping area could be a penthouse, or a small part of a next-door building, and, therefore, the match will be flawed. Calculating the size of the overlapping areas between the polygons already available and the ones detected is a second approach. Then, the value assigned to the initial polygon will be the value of the detected polygon, with which the first shares the largest common area.

During the current project we manually measured 100 buildings' heights in order to evaluate the results of the automatic detection, in lack of original data. Openstreetmap was updated with that information concerning the field "levels" in those buildings. Further update, automatically, is yet under study, since it requires a high accuracy of detection, combined with a high accuracy in determining the levels of each building and assigning it to the original data.

4. EVALUATION

The final results were evaluated based on the polygons available on openstreetmap.org. The results of a small part of the study area are presented in Figure 14.

Figure 14: Openstreetmap polygons (blue) compared to the detected polygons (brown)

The polygons were evaluated as to the number of detected polygons compared to the total existing building. As shown in Figure 14, due to various reasons (chapter 3.4) openstreetmap polygons appear larger than the ones detected by the algorithm. Openstreetmap dataset is created by users and not experts, and, therefore, the digitization of the buildings is not always accurate in shape and size. Detections that were not buildings were marked as "false alarms" (Table 1).

Table 1

	Total area	**Smooth terrain**	**Intense terrain**
Number of detected polygons	84%	93.50%	83%
False alarms	1.50%	0.50%	4%

In order to determine the reasons of non- detection, we also evaluated the results between the stages of the entire process. False alarms were reduced when detecting individual buildings, and were almost zeroed after the GIS processing. Eliminating the false alarms and removing the elongated parts of the polygons, though, had a high cost in the detection's results. The percentages are shown in Table 2.

Table 2

Stage of the procedure	**Detection percentage/stage**	**False alarms**
Threshold in elevation from DTM	93%	40%
Detecting building levels in complexes	91%	20%
GIS processing	84%	1.50%

Finally, based on the manually measured buildings, we evaluated the automatically calculated levels of the buildings, comparing the two values. Due to rounding the floor- numbers to the closest integer, even a small deviation in height (e.g. of 0.20 meters) may lead to one floor's difference. Only 4% of the buildings were totally miscalculated, especially because of poor description of the terrain (Table 3).

Table 3

Zero deviation	75%
Deviation of 1 floor	21%
More than 2 floors	4%

Results regarding the entire study area are presented in Figure 15, where the city planning model of Athens is reflected as previously described. Besides updating openstreetmap.org, the algorithm provides a first overall picture of the density and the height of all buildings in the city. Visualized by a graded palette from blue to red, totally 16.600 buildings were detected and mapped, as shown in the following picture:

Figure 15: Polygons of buildings detected in the entire study area; higher (red) buildings are located around major road axes, whereas smaller (blue) buildings are mostly found in residential areas

5. CONCLUSION

The detection process combines photogrammetric and remote-sensing methods with Geographical Information Systems. Using only aerial images, with no further data required it detects the major percentage of buildings for a large study area, low cost and time. Further improvement of the detection procedure (by using images in larger scale, different focal length, etc.), combined with additional available data for the area (DTM, LIDAR data) may lead to even higher detection rates along with higher accuracy in determining the levels of the buildings. Merging the datasets with already existing open source databases requires both an improvement on the detection algorithm and an effective GIS process to avoid data loss and false-updates.

REFERENCES

Balenović, I., Marjanović, H., Vuletić, D., Paladinić, E., Sever, M., Indir, K., 2016. Quality Assessment of High Density Digital Surface Model over Different Land Cover Classes, *Periodicum Biologorum*, vol. 117(4).

Dall'Asta, E., Roncella, R., 2014. A Comparison of Semiglobal and Local Dense Matching Algorithms for Surface Reconstruction. In: *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 40(5): 187.

Ernst, I., Hirschmüller, H., 2008. Mutual Information Based Semi-Global Stereo Matching on the GPU. In: *International Symposium on Visual Computing*, Lecture Notes in Computer Science, vol. 5358, pp. 228–239.

Haala, N., 2013. The Landscape of Dense Image Matching Algorithms. In: *Photogrammetric Week 2013*, Eds.: Dieter Fritsch, pp. 271-284.

Kiran, S.S., 2015. Three-Dimensional City Planning Using Photogrammetry and GIS. *Directions Magazine*.

Lin, C., Nevatia, R., 1998. Building Detection and Description from a Single Intensity Image. *Computer Vision and Image Understanding*, vol. 72(2): 101–121.

Liu, W., Prinet, V., 2005. Building Detection from High-Resolution Satellite Image Using Probability Model. In: *Proceedings of Geoscience and Remote Sensing Symposium, IGARSS'05, IEEE International*, 6:3888–3891.

Müller, S., Zaum, D.W., 2005. Robust Building Detection in Aerial Images. In: *International Archives of Photogrammetry and Remote Sensing*, vol. 36 (B2/W24): 143–148.

Rottensteiner, F. 2003. Automatic Generation of High-Quality Building Models from Lidar Data. *IEEE Computer Graphics and Applications*, vol. 23(6): 42–50.

Rottensteiner, F., Trinder, J., Clode, S., Kubik, K., 2007. Building Detection by Fusion of Airborne Laser Scanner Data and Multi-Spectral Images: Performance Evaluation and Sensitivity Analysis. *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 62(2): 135–149.

Tarsha-Kurdi, F., Landes, T., Grussenmeyer, P., 2008. Extended RANSAC Algorithm for Automatic Detection of Building Roof Planes from LiDAR Data. *The Photogrammetric Journal of Finland*, vol. 21(1): 97–109.

Vosselman, G., Dijkman, S., 2001. 3D Building Model Reconstruction from Point Clouds and Ground Plans. In: *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, vol. 34 (3/W4): 37–44.

Zhang, Y., 1999. Optimisation of Building Detection in Satellite Images by Combining Multispectral Classification and Texture Filtering. In: *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 54(1): 50–60.

Cohen, J.P., Ding, W., Kuhlman, C., Chen, A., Di, L., 2016. Rapid Building Detection Using Machine Learning. *Applied Intelligence*, vol. 45(2): 443–457.

Vakalopoulou, M., Karantzalos, K., Komodakis, N., Paragios, N., 2015. Building Detection in Very High Resolution Multispectral Data with Deep Learning Features. In: *Geoscience and Remote Sensing Symposium (IGARSS), 2015 IEEE International*, pp. 1873–1876.