AUTOMATIC MODELLING OF 3D TREES USING AERIAL LIDAR POINT CLOUD DATA AND DEEP LEARNING

R.G. Kippers[1], L. Moth[2], S.J. Oude Elberink[3]

[1] Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS), University of Twente - r.g.kippers@student.utwente.nl
[2] Deltas, Coasts and Rivers, Witteveen+Bos - luke.moth@witteveenbos.com
[3] ITC , Faculty of Geo-Information Science and Earth Observation, University of Twente - s.j.oudeelberink@utwente.nl

KEY WORDS: AHN, Aerial Laser Scanning, Point Cloud, PointNet, Deep Learning, CityJSON

ABSTRACT:

3D tree objects can be used in various applications, like estimation of physiological equivalent temperature (PET). During this project, a method is designed to extract 3D tree objects from a country-wide point cloud. To apply this method on large scale, the algorithm needs to be efficient. Extraction of trees is done in two steps: point-wise classification using the PointNet deep learning network, and Watershed segmentation to split points into individual trees. After that, 3D tree models are made. The method is evaluated on 3 areas, a park, city center and housing block in the city of Deventer, the Netherlands. This resulted into an average accuracy of 92% and a F1-score of 0.96.

1. INTRODUCTION

How can urban plans be made to adapt cities to be more resilient to extreme temperatures? For finding critical ‘hot spots’, a temperature perception heat map can be used. Witteveen+Bos developed such a heat map for the Netherlands, based on the physiological equivalent temperature (PET). Trees can impact temperature perception by evaporating water, providing shade, or impeding a fresh breeze. Therefore, the current PET-model utilizes a tree map. This tree map only contains 2D information on the presence of a tree, on a grid with squared tiles of 50 cm. A more detailed tree map could improve the accuracy of the PET-model.

The AHN (Actueel Hoogtebestand Nederland) is an airborne data set available in 2D and 3D, for The Netherlands. The 3D data consists of a point cloud, which is a list of Cartesian (X,Y,Z) coordinates together with features. The average point density of the data source is 8 points per m^2. A detailed overview of the data set can be found on the website[1]. Figure 1 shows a subset of the AHN version 3 (AHN-3). The AHN dataset is used as input source for this project.

There is a variety of methods available for point cloud data analysis. A short summary will be presented in section 2. Recent literature uses deep learning methods to extract information from point clouds. Commonly used deep learning architectures are PointNet (Qi et al., 2017a) or other variations. These networks observe patterns in labeled point cloud data and can use these learned patterns to label unseen data.

This paper introduces a simple method to create 3-dimensional tree models using an areal LIDAR point cloud. The novelty of this method is the ability to deploy this method on a large scale, e.g. for an entire country, in a cloud computing environment. The method uses relatively few resources and can therefore run on a large scale in Microsoft Azure Batch on entry level virtual machines.

[1] *https://www.ahn.nl/kwaliteitsbeschrijving*

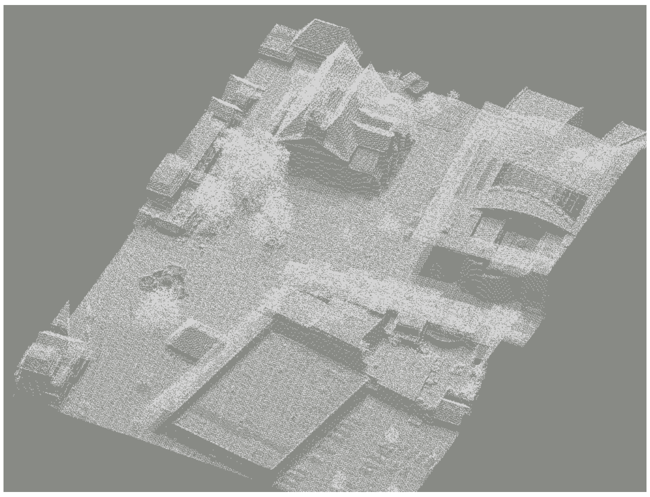
Figure 1. Screenshot subset AHN

2. RELATED WORK

Point cloud processing consists of a variety of tasks. A few examples are height measuring, object reconstruction, mesh generation and object detection. Traditional approaches of point cloud processing make use of rule-based point cloud manipulation techniques like feature based filters, decision trees or machine learning approaches such as support vector machine. They also make us of spatial characteristics by using algorithms like Density-based spatial clustering of applications with noise, DBSCAN (Ester et al., 1996) (or variations like (Wang et al., 2019)). Point cloud data is irregular. To get rid of this irregularity, sometimes point cloud processing is applied on voxel point clouds, in which a voxel represents a certain area in the 3-dimensional space (Poux and Billen, 2019). In 2017, Qi et. al. introduced PointNet (Qi et al., 2017a), a deep learning network architecture that can be adjusted to fit multiple applications, such as object classification, part segmentation and semantic segmentation. PointNet++ (Qi et al., 2017b) can process more fine-grained patterns and complex scenes. (Song et al., 2020) explains how points of segmented objects can be identified using CNN's and Hough transformations. Hough transformations are a commonly used used feature extraction method in the 2D computer vision domain (Cantoni and Mattia, 2013). A comprehensive comparison of deep learning networks regarding point cloud processing can be found in (Organokov and team, 2020).

(Meijer et al., 2015) uses areal images from AHN-2 to identify trees and converts them into Silvi-Star ((Koop, 1989)) models. This model describes a tree based on the X,Y,Z coordinates of 8 points; the trunk base (B), height of the first leaves (F), top of the crown (T), bottom of the crown (C) and 4 points of the circumference ($P1-P4$). Modeling a tree in the Silvi-Star method enables further calculations. (Lucas et al., 2019) uses the point cloud from the AHN-3 to identify trees in rural areas. It does so by first classifying single points, and after that segmenting points by surface growing. Literature by Soilán et. al. (Soilán et al., 2019) tried to reproduce the AHN classification (ground, vegetation and buildings) by using PointNet. The PointNet network was trained on a segment of 6.25x5 km^2. The researchers used diferent combinations of targets and features. The model accuracy was promising, but there was still a high confusion between vegetation and building class in the models.

Segmentation, the process of grouping points that belong to an individual object, can be done in a variety of ways. Xiao et al. (Xiao et al., 2016) clustered points belonging together using mean shift algorithm and the Pollock model (Pollock, 1994). The Pollock model obtains the shape of a tree crown and by changing its parameters a, b, the model can obtain multiple tree crown shapes. The equation can be found below. The Watershed algorithm starts at predefined markers (e.g. tree tops) and increases the area until it reaches the area of another cluster. This is done by filling metaphorically 'basins'. Watershed is not applied on the direct points, but on a Canopy Height Model (CHM). The algorithm has a linear time complexity (Kornilov and Safonov, 2018).

$$P = \frac{z^n}{c^n} + \sqrt{(\frac{x^2}{a^2} + \frac{y^2}{b^2})^n} = 1 \tag{1}$$

3. METHOD

One of the requirements of this method is speed, so it is able to run for large areas. The approach used in this project is first classification and after that object segmentation. First, for each point a class is predicted, to only keep points belonging to a tree. After that, all tree-type-points will be segmented into individual trees.

3.1 Classification

For classification, the PointNet architecture is used. This architecture is shown in Figure 2. The following two setups are compared: a 2-class PointNet network and a 3-class network. The first set-up has targets 'other' (0) and 'tree' (1), the second has the targets 'other', 'tree' and 'building' (2). The deep learning network is implemented in Keras, using the Adam optimizer and weighted categorical cross entropy as loss function. A weighted loss function is chosen since the dataset is unbalanced. Depending on the area, the class 'other' consists often over 80% of the points. Equation 3 shows the used formula, where M are the number of targets, and w_c corresponds to the target class weight.

$$-\sum_{c=1}^{M} y_{o,c} \log(p_{o,c}) w_c \tag{2}$$

The weight constants are determined based on the training data. To create training data, manually annotated 2D shape files of areas are combined with the AHN point cloud. The labeled data set consists of 11,550 samples of 2048 points. Of all samples, 60% is for used for training, 20% for validation and 20% for testing. During training, the Z-axis is replaced by the height above ground, calculated by the Height Above Ground Delaunay filter in PDAL (Contributors, 2018). The PointNet architecture needs a fixed amount of point as input. In this setup, this number (2048) is chosen based on the memory constraints of the GPU. The number of trainable parameters in Keras is 891,855. Because of the fixed input, the dataset is split into tiles of 2048 points. This is done by fixing the y-axis and moving the x-axis until the tile contains approximately 2048 points.

Name	Network	Features	Targets
S1	PointNet	X,Y,Z,AhnClass,NumReturns	0,1,2
S2	PointNet	X,Y,Z,AhnClass,NumReturns	0,1

Table 1. Neural Networks

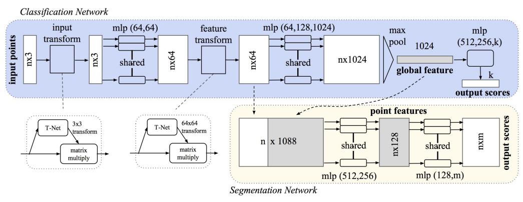

Figure 2. PointNet architecture (Qi et al., 2017a)

3.2 Segmentation

Two different segmentation algorithms are used in this project, Watershed segmentation and Mean shift segmentation. One of these will be used in the final process. The first, watershed, is applied on a Canopy Height Map (CHM). A cell in this CHM with edges of 0.25m. A Gaussian filter is applied on the CHM to make it less sensitive to outliers. For this algorithm, markers needs to defined before segmenting. In this project, these markers will represent tree tops. Cells are marked as tree top if they are the highest point in the following radius:

$$Radius = max(4, 4 + \frac{1}{8}(z - 10)) \tag{3}$$

During Mean Shift segmentation, in each step, points converges towards the center of a kernel. As described in the related work section, the Pollock kernel can be used. The main idea is to move all points belonging to the same tree to the crown center.

3.3 Tree modelling

Two different methods of converting points to a 3D-model have been tested: Pollock model fitting using least squares ((Xiao et al., 2016)) and finding maximum points in a direction on a slice. For both methods, first the trunk will be removed, using the method in (Xiao et al., 2016). To find the exterior of a tree, the following steps are executed:

1. Calculate the convex hull using the Python library SciPy.

2. Project a random distribution on the vertices of the convex hull. In this way, the points on the exterior are evenly distributed.

Mean Shift using Pollock Research by (Xiao et al., 2016) shows how a tree crown can be cut in half on the median height. An example can be found in figure 3. After that, a Pollock model can be fitted on each side using least squares.

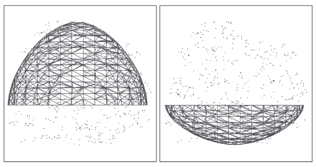

Figure 3. Pollock model fitting (Xiao et al., 2016)

Slicing A simple model of converting a tree into a 3-dimensional model is slicing. The tree get sliced at 0%, 25%, 50%, 75% and 100% of its height. The dimensions on each slice are used as points in a 3D model. On the 25-75% slices, a octagon is made by finding maximum points. In this way, for the crown just 26 points needs to be stored, and the crown shape will be kept.

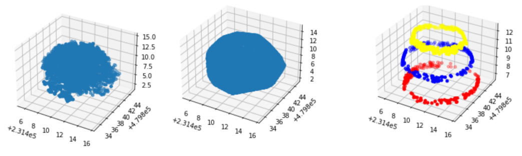

Figure 4. Points, random dist. points, slices

3.4 Implementation

The code is optimized to run on Microsoft Azure Batch. in this way, multiple instances can run simultaneously on a highly scalable platform.

4. RESULTS

In this section, the results of the methods proposed in the previous section will be presented.

4.1 Classification

As mentioned the previous section, two different deep learning architectures are tested. Training the deep learning network is done using 6900 samples. A confusion matrix for each setup is presented in table 2 and 3. The rows represent true labels, columns represent predictions. The setup S1, using 3 targets, has difference of 1% less misclassified points as tree, but its true positive rate is equal to the positive rate for the two-target network. The 3-target network is kept for the final process.

Other	0.91	0.09	0
Tree	0.04	0.95	0
Building	0.01	0	0.99
	Other	**Tree**	**Building**

Table 2. S1 - PointNet

Other	0.72	0.28
Tree	0.05	0.95
	Other	**Tree**

Table 3. S2 - PointNet 2 class

4.2 Segmentation

In the current setup, the Mean Shift method resulted in a lot of noise generated by outliers. Points that do not fit in a single valid Pollock kernel create new objects, therefore not resulting in a proper tree object. This can be for points in the x, y as well on the z-axis. Also, the algorithm can sometimes be slow to converge. Two variants where tested: one based on all points and one based on only the top points, using a CHM.

For Watershed segmentation, two methods where tested to detect markers. The markers are found based on the CHM and on the actual points. For both methods, the same radius is used. After the implementation, it became clear that the CHM based approach is less sensitive to outliers, but gives in situations with a high density of trees a more precise result. But both give a reasonable result. The output is not compared with a ground truth. Since the CHM-based approach is also computational less expensive (in both space and time complexity), this approach is chosen.

4.3 Tree modelling

Using the Least Squares algorithm did result in a few extreme values. Therefore, this method was, without adding constraining boundaries, not usable. The 'split' algorithm did not result in any problems. Since the trunk is not of interest for this project, the trunk has a fixed diameter.

5. EVALUATION

5.1 Final process

The process which outputs the data for the evaluation, consists of the following components. First, the 3-class PointNet network is used. After that, the Watershed algorithm using the CHM for markers is applied. Finally, the tree is modelled using the 'slice' technique and stored in CityJSON. CityJSON (Ledoux et al., 2019) is a JSON-based encoding for the storage of digital 3D city models. CityJSON consists of multiple city objects, such as buildings, bridges and solitary vegetation. The latter is used as tree object type. Since CityJSON supports multiple Level of Details (LoDs), a tree is stored in 3 different LoDs: a 2D octagon, a 2D alphashape and a 3D model. All LoDs can be found in Figure 5. Figure 6 shows the 3D shapes compared to images from Google Streetview. In Figure 7, the method 3D result is shown for a few streets in the city center of Amsterdam.

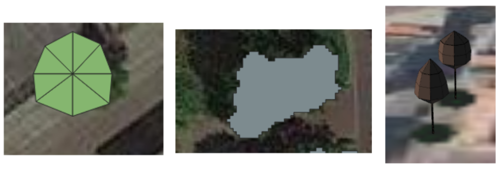

Figure 5. Different tree representations (octagon, alphashape, 3D)

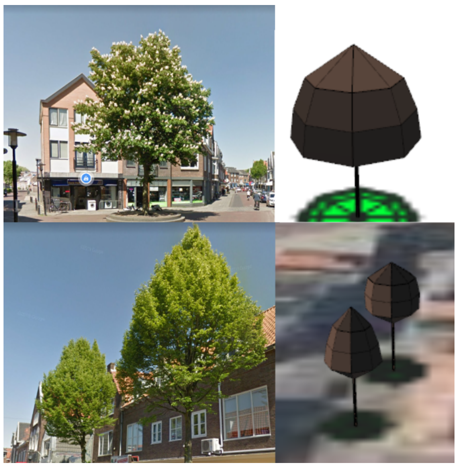

Figure 6. Google Streetview (left) and 3D Tree (right)

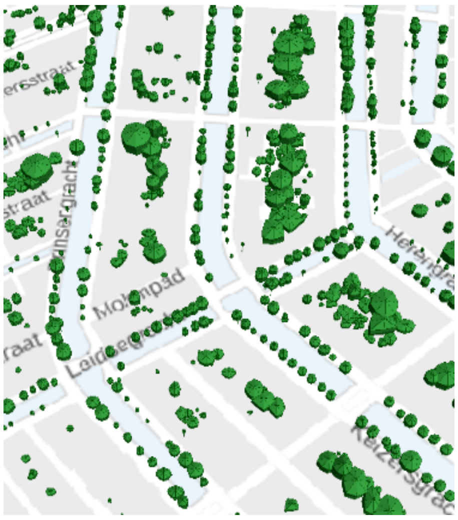

Figure 7. Amsterdam City Center

5.2 Area selection

The outcome of the algorithm is assessed on 3 different areas; city center, a park and a housing block. For each area, the outcome is compared with the aerial images available on Google Maps.

5.3 City Center

The selected city center area is a typical historical city center town square, with older houses on each side of the square.

True positives	67
False positives	4
False negatives	1
Segmentation error	2

Table 4. Result city center

In the city center part, a few things are not correct. During point-wise classification, two trucks are wrongly classified as tree, as can be seen in Figure 9. However, since they have a much lower height (top is 3.5 meters), they can be easily filtered out of the dataset in a GIS application like QGIS. Due to the small distance between trees, two tree tops are not found, and therefore these trees will not be seen as individual trees. This can be seen on the canopy height model in Figure 10.

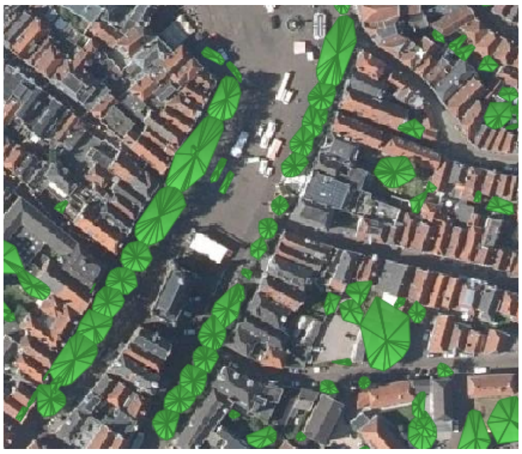

Figure 8. Result City Center

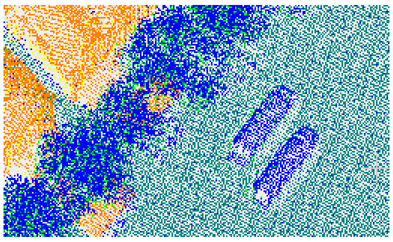

Figure 9. Trucks classified as tree

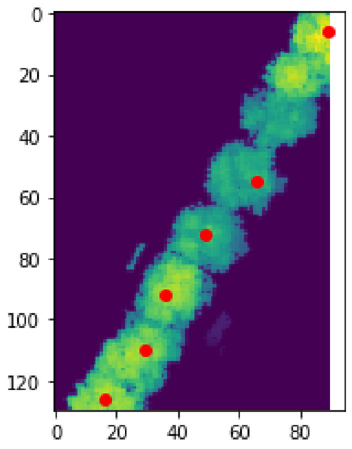

Figure 10. Missed tree top in CHM

5.4 Park

The Rijsterborgherpark in Deventer, is a park created in 1874. Its location is close to the main train station, and one side is bounded by a train track. A lot of it was destroyed during the second world war, therefore the park was renovated in 1954. Still, a lot of old trees can be found in the park.

Some of the overhead line of the train track on the edge of the park gets wrongly classified as tree. Since a lot of trees are clustered together, which makes it difficult to determine individual trees by hand, the segmentation error score is not calculated for this area. The result can be found in table 5.

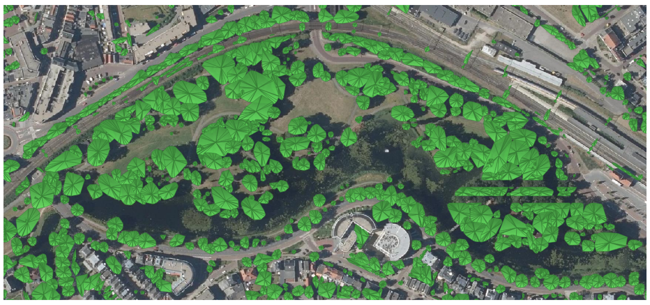

Figure 11. Park tree map 2D

True positives	214
False positives	3
False negatives	6

Table 5. Result park

5.5 Housing block

The selected housing block represents an older social housing area with houses built around 1960. All houses have a backyard, most of them with a shed. The backyards contain a variety of vegetation.

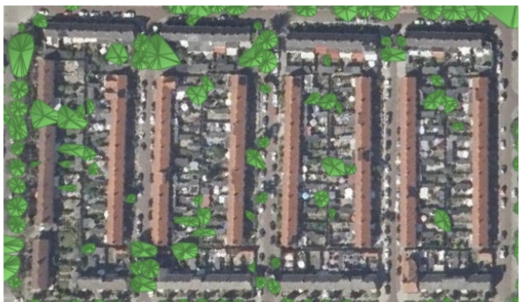

Figure 12. Housing block tree map 2D

It was difficult to score this area, since in the back yards, it is often unclear if a piece of vegetation is a tree, or belongs to something else, like a hedge. However, some trees in the backyard are not in the tree map. These trees are narrow and low, and therefore have few points. The points were assigned the right target during classification but did not form a tree during segmentation.

True positives	67
False positives	2
False negatives	6
Segmentation error	1

Table 6. Result housing block

5.6 Computing time

To run the full algorithm on an AHN tile (5km x 6.25 km), about 29 hours of processing on a server with 4 gigabytes and 2 computing cores is needed. Using Azure Batch, this task can run in parallel on multiple machines. This includes downloading and unzipping LAZ files.

6. CONCLUSION

In this project, a tree map is created using a Keras Tensorflow Deep Learning network (PointNet (Qi et al., 2017a)) and segmentation using the Watershed algorithm. Especially in areas with a high tree density, tree segmentation is a difficult task, that does not generalize well from tree dense areas to e.g. city centers. As can be seen from the results in the result assessment chapter, the output of this algorithm is not perfect, but most of the trees are correctly classified. The method resulted into an average accuracy of 0.92. The average precision is 0.96 (city center = 0.94, park = 0.99 and housing = 0.97) and the average recall is also 0.96 (city center = 0.99, park = 0.98 and housing = 0.96). Since both the precision and recall is 0.96, the F1-score is 0.96. Processing an area of 31.25 km^2 takes about 29 hours.

Since the labeled data set consists of 11.550 samples in only a few locations, classification of unseen objects like train overhead lines might induce errors. However, since the output also contains attributes like the number of points in a tree tree height and crown height, it is possible to utilize GIS applications to remove these false positives using filters.

The metrics of the PointNet network could possibly be improved by augmenting the data set with more data. This can for example be infrared, or the pixel color, obtained from other sources. This would make the network less sensitive to unseen objects.

ACKNOWLEDGEMENTS

This project was conducted in 2020, during an internship at Witteveen+Bos.

REFERENCES

Cantoni, V., Mattia, E., 2013. *Hough Transform*. Springer New York, New York, NY, 917–918.

Contributors, P., 2018. Pdal point data abstraction library.

Ester, M., Kriegel, H.-P. et al., 1996. A density-based algorithm for discovering clusters in large spatial databases with noise.

Koop, H., 1989. Forest dynamics, SILVI-STAR : a comprehensive monitoring system. PhD thesis.

Kornilov, A. S., Safonov, I. V., 2018. An Overview of Watershed Algorithm Implementations in Open Source Libraries. *Journal of Imaging*, 4(10). https://www.mdpi.com/2313-433X/4/10/123.

Ledoux, H., Ohori, K. A., Kumar, K., Dukai, B., Labetski, A., Vitalis, S., 2019. CityJSON: A compact and easy-to-use encoding of the CityGML data model. *Open Geospatial Data, Software and Standards*, 4(1), 4.

Lucas, C., Bouten, W., Koma, Z., Kissling, W., Seijmonsbergen, A., 2019. Identification of linear vegetation elements in a rural landscape using LiDAR point clouds.

Meijer, M., Rip, F., van Benthem, R., Clement, J., van der Sande, C., 2015. Boomkronen afleiden uit het actueel hoogtebestand nederland: kwaliteitsaspecten rondom het geautomatiseerd in kaart brengen van bomen op basis van het ahn2-bestand. Technical report, Alterra, Wageningen-UR.

Organokov, M., team, D., 2020. Data study group final report: Sensat semantic segmentation of 3d point clouds.

Pollock, R. J., 1994. Model-based approach to automatically locating tree crowns in high spatial resolution images. J. Desachy (ed.), *Image and Signal Processing for Remote Sensing*, 2315, International Society for Optics and Photonics, SPIE, 526 – 537.

Poux, F., Billen, R., 2019. Voxel-based 3D point cloud semantic segmentation: unsupervised geometric and relationship featuring vs deep learning methods. *ISPRS International Journal of Geo-Information*, 8(5), 213.

Qi, C. R., Su, H., Mo, K., Guibas, L. J., 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 652–660.

Qi, C. R., Yi, L., Su, H., Guibas, L. J., 2017b. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. *CoRR*, abs/1706.02413. http://arxiv.org/abs/1706.02413.

Soilán, M., Lindenbergh, R., Riveiro, B., Sánchez-Rodríguez, A., 2019. POINTNET FOR THE AUTOMATIC CLASSIFICATION OF AERIAL POINT CLOUDS. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2/W5, 445–452. https://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/IV-2-W5/445/2019/.

Song, W., Zhang, L., Tian, Y., Fong, S., Liu, J., Gozho, A., 2020. CNN-based 3D object classification using Hough space of LiDAR point clouds. *Human-centric Computing and Information Sciences*, 10, 1-14.

Wang, C., Ji, M., Wang, J., Wen, W., Li, T., Sun, Y., 2019. An improved DBSCAN method for LiDAR data segmentation with automatic Eps estimation.

Xiao, W., Xu, S., Oude Elberink, S., Vosselman, G., 2016. Individual tree crown modeling and change detection from airborne lidar data. *IEEE Journal of selected topics in applied earth observations and remote sensing*, 9(8), 3467–3477.