AUTOMATIC CLASSIFICATION OF TREES FROM LASER SCANNING POINT CLOUDS

Beril Sirmacek, Roderik Lindenbergh

Department of Geoscience and Remote Sensing, Delft University of Technology, Stevinweg 1, 2628CN Delft, The Netherlands
(B.Sirmacek, R.C.Lindenbergh)@tudelft.nl

Commission I/3

KEY WORDS: Point Clouds, Tree Detection, Classification, Terrestrial Laser Scanning, Mobile Laser Scanning, Airborne Laser Scanning

ABSTRACT:

Development of laser scanning technologies has promoted tree monitoring studies to a new level, as the laser scanning point clouds enable accurate 3D measurements in a fast and environmental friendly manner. In this paper, we introduce a probability matrix computation based algorithm for automatically classifying laser scanning point clouds into 'tree' and 'non-tree' classes. Our method uses the 3D coordinates of the laser scanning points as input and generates a new point cloud which holds a label for each point indicating if it belongs to the 'tree' or 'non-tree' class. To do so, a grid surface is assigned to the lowest height level of the point cloud. The grids are filled with probability values which are calculated by checking the point density above the grid. Since the tree trunk locations appear with very high values in the probability matrix, selecting the local maxima of the grid surface help to detect the tree trunks. Further points are assigned to tree trunks if they appear in the close proximity of trunks. Since heavy mathematical computations (such as point cloud organization, detailed shape 3D detection methods, graph network generation) are not required, the proposed algorithm works very fast compared to the existing methods. The tree classification results are found reliable even on point clouds of cities containing many different objects. As the most significant weakness, false detection of light poles, traffic signs and other objects close to trees cannot be prevented. Nevertheless, the experimental results on mobile and airborne laser scanning point clouds indicate the possible usage of the algorithm as an important step for tree growth observation, tree counting and similar applications. While the laser scanning point cloud is giving opportunity to classify even very small trees, accuracy of the results is reduced in the low point density areas further away than the scanning location. These advantages and disadvantages of two laser scanning point cloud sources are discussed in detail.

1 INTRODUCTION

Trees do vital job for well-being of all species on the planet. They do not only increase beauty, prove food and living space for animals, but they also absorb carbon dioxide and give oxygen which contributes to the environmental circulations. Therefore, for scientists, for municipalities and other governmental agencies, it is important to track the numbers of trees and to measure their physical attributes for scientific analysis such as carbon dioxide absorbing volume. Laser scanning technology provides opportunity to do all these physical measurements in a computer environment without disturbing the nature and by spending less man effort.

Most tree classification studies in literature focus on separating trees in environments where there is little variation in objects. Therefore, separating tree points problem is solved by removing the ground points and checking the appearance of tree trunks or tree crowns. When the point cloud is from an urban region, it might contain points of many different objects such as building facades, cars, light poles, traffic signs, etc. In this case, algorithms need to segment the point clouds of different objects in order to control their 3D shapes. By controlling shapes, the algorithms decide if the points are coming from a tree or another object. These algorithms generally depend on either voxel space creation or fitting pre-defined geometrical models to the point clouds. The voxel space creation based methods are generally very sensitive to the voxel size selection. The methods which are using geometrical models also need specific parameter selection and they have a risk of missing trees which have trunk shapes different from the pre-defined models.

Some of the previously introduced approaches for processing regions having only trees as objects are as follows. Pyysalo and Hyyppa (2002) introduced a method to detect individual trees in dense forest point clouds which are acquired by an airborne laser scanner. They have used a Digital Terrain Model (DTM) to remove ground, which means that additional data is required to process the point clouds. Lalonde et al. (2006b) proposed a tree classification approach from terrestrial laser scanning point clouds by detecting the ground surface. The method worked using a statistical classification technique which uses linear-ness and scatter-ness of local area. The method had some disadvantages such as difficulty of scale selection, high computation time and difficulty of noise removal. Rutzinger et al. (2010) segmented their point cloud into homogeneous planar regions and they removed non-vegetation segments, which are large and planar. The remaining laser echoes are re-labelled by grouping nearby points to connected components. For these components the roughness (standard deviation of elevation) and a point density ratio in a given height interval are calculated. They observed that the point density ratio is significantly low for trees because the tree crown intercepts much more echoes than the tree trunk. Therefore, trees are extracted by selecting connected components with high roughness and low point density ratio.

McDaniel et al. (2012) have developed an automatic individual tree detection method using terrestrial laser scanning point clouds. They have started with ground plane segmentation. After fitting a ground surface, they have estimated breast height for trees and detected tree trunks. Finally, they have used the K-means classification method to decide for the rest of the points if they belong to a tree trunk. They have tested the algorithm in different environments with different appearance. They have observed that the more straight the ground surface the better performed the algorithm. Kaartinen et al. (2012) have selected nine different tree classification methods from literature and tested them on a for-

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
Editors: S. Oude Elberink, A. Velizhev, R. Lindenbergh, S. Kaasalainen, and F. Pirotti
doi:10.5194/isprsannals-II-3-W5-137-2015

est laser scanning point cloud. They have discussed strength and weaknesses of the different methods in terms of four scientific criteria; number of detected trees, accuracy of the detected tree locations, accuracy of the detected tree heights and crown delineation accuracy. They have also noticed that for in all nine different methods, higher trees are detected more accurately than lower ones.

Bremer et al. (2013) separated objects in the point clouds by connected component and Dijkstra-path analysis. They have classified trees using a graph based approach considering the branching levels of the given geometries. Gorte and Pfeifer (2004) have generated a range image in order to segment trees. After segmenting individual trees, they mapped the points back to a voxel space, where it is segmented into branches using mathematical morphology. The voxel size selection has played a very crucial role in well performance of the algorithm. Tittmann et al. (2011) introduced a RANSAC approach to fit geometric models on point clouds in order to identify individual tree crowns. Li et al. (2012) introduced a method to segment individual trees in dense forest point clouds. The method depends on detecting seed points (local maximums) which are assumed as tree tops and fitting a cone shape from the seed points to the ground in order to segment the point clouds. The proposed algorithm provided good results even in dense coniferous forests. However, they have indicated that the algorithm might not be applicable to forests which consist of other types of trees. Besides, the algorithm failed to detect trees which are very short or which have low point density.

Pu et al. (2011) segmented the point cloud and checked the 3D geometries of the segments. If an object has a cylindrical body and a large crown on the top, then it is labelled as a tree. The trees are not detected if their cylindrical shape trunks are not visible or if their crown diameter is not large enough. In the second case, the trees in a street point cloud might end up being labelled as poles. Monnier et al. (2012) proposed a tree detection algorithm based on local geometric descriptors computed on each laser point using a fixed neighbourhood. These descriptors described the local shape of objects around every 3D laser point. A projection of these values on a 2D horizontal accumulation space followed by a combination of morphological filters helped them to detect individual tree clusters. Lalonde et al. (2006a) have extracted 3D features from the point clouds to classify points as trees and non-trees. They have then applied a segmentation algorithm to identify individuals. Finally they have used 3D primitive geometric shapes such as cylinders in order to generate 3D tree models. They have used the models to estimate the breast hight diameter on ground and also aerial point clouds. Raumonen et al. (2011), proposed a method for automatically extracting approximate tree branch measures from point clouds. The method assumed that the tree can be locally approximated with cylinders without using voxel spaces. It performed well for detecting tree trunks and relatively large branches. However, it could not show good performance on detecting thin, irregular shaped, or low-point-density branches. Bucksch et al. (2009) detected individual trees by using a region growing approach starting from seed points. They have used individual trees for breast height diameter estimation after skeletonizing the point clouds.

We have noticed that these graph generation, model matching and 3D feature extraction based methods not only need considerable amount of processing time, but they are also not able to detect a tree when it has a different 3D geometry than the trees which are used for training the algorithm parameters.

The literature survey shows us that; most of the previously proposed approaches have complex implementation steps, they generally need high computation time and accurate parameter definition plays an important role in order to obtain successful results. We believe that, there is still a need for fast and reliable tree classification algorithms for processing large data sets which include trees but also many other types of objects. Herein, we propose an automatic approach to classify tree points in laser scanning point clouds. The classification is done by generating a 2D probability matrix which is an imaginary surface approximately on the ground layer level. This probability matrix highlights the possible tree trunk locations and it helps to identify if a point belongs to a tree. The resulting point cloud basically has the same x, y, z coordinates as the input points. However it contains one extra column which indicates if the point belongs to the 'tree' class or 'non-tree' class. We test the algorithm on mobile laser scanning (MLS) and airborne laser scanning (ALS) point clouds.

2 TREE CLASSIFICATION

Before identification of individual trees for further detailed analysis, an important and helpful step is to classify the tree points in the input point cloud. This classification basically refers to assigning a class value 1 to the points which belong to a tree and a class value 0 to the points which belong to other objects in the point cloud. Once tree points are separated from points sampling other objects, it is easer to identify individual trees and making necessary measurements on them.

For classifying the points in the input point cloud as tree points, we introduce a new approach based on the following steps;

1- Generating a probability matrix
2- Selecting maxima which indicate the locations having high probability to have a tree trunk
3- Assign points to the 'tree' or the 'non-tree' class
4- Filter the ground points
In the following parts, we explain each step in detail.

2.1 Generating A Probability Matrix

Our classification method works in 3D space and keeps the original locations of the 3D points in the cloud. However, at the processing stage, we use a 2D matrix (which we will call the $V(x, y)$ probability matrix) for making a decision about the approximate locations of the tree trunks. This 2D matrix is an imaginary grid in the (x, y) plane, appearing between (X_{min}, Y_{min}), (X_{min}, Y_{max}), (X_{max}, Y_{min}) and (X_{max}, Y_{max}) geographical coordinates. Here, X_{min}, X_{max} and Y_{min}, Y_{max} correspond to the minimum and maximum x, and y coordinate values of the input points. The grid step sizes of the $V(x, y)$ matrix are determined by considering the ground sampling resolution of the point cloud and the diameters of the smallest trees (0.5 meters in our examples) that we would like to be able to classify correctly.

We introduce the steps of the algorithm on a tree point cloud example given in Figure 1.(a) which shows a mobile laser scanning point cloud from a view angle perpendicular to the (x, z) plane. The same point cloud is shown in Figure 1.(b) having false colors corresponding to the height values of the points. Figure 1.(c) shows the calculated $V(x, y)$ matrix which is also false colored in order to show the differences more clearly.

The pseudocode given in Figure 2 presents the algorithm that we use to compute and to assign values to the grids of the $V(x, y)$ matrix. The $V(x, y)$ matrix is filled with zero values at each grid as an initial value. Here P_{num} is equal to the total number of points in the input point cloud P. The symbols $P(i).x$, $P(i).y$, $P(i).z$ stand for the x, y, z coordinates of the i-th point of the

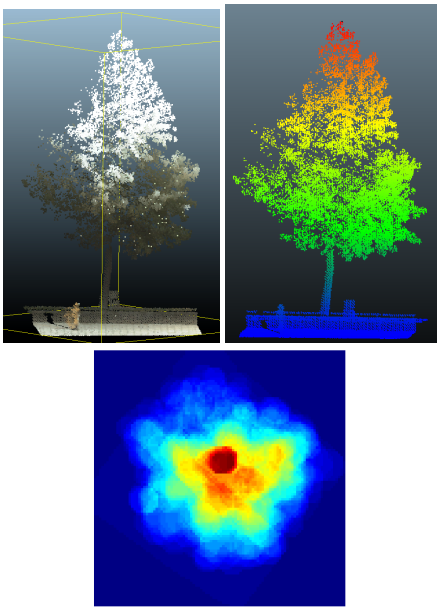

Figure 1: (a) An example tree point cloud from a mobile laser scanning data, (b) The same tree point cloud with height values assigned to the points as false color, (c) The calculated $V(x, y)$ matrix is shown with false colors where the red color corresponds to the highest and the blue color corresponds to the lowest value.

point cloud P respectively. The X and Y variables indicate the position of a point in the 2D grid. Finally, the x_{step} and y_{step} variables stand for the grid size in X and Y dimensions respectively. In this study, we choose x_{step} equal to y_{step}. The value 0.3 (in meters) is assigned to these variables as grid step size in our example.

If the grid step size is very large then we can miss some trees which do not have a large diameter, if the step size is too small then we might use unnecessarily large memory space. Therefore, the user must decide to set the grid step size wisely considering the diameter of the smallest tree which is interesting for the application.

2.2 Selecting Local Maxima Points

The 'for' loop in the pseudocode given in Figure 2, basically adds the height of every point into the corresponding grid. We expect the $V(x, y)$ matrix to have very high values at the positions where tree trunks are located, since there will be many trunk points which correspond to the same grids of the (x, y) plane. Therefore after generating the $V(x, y)$ matrix, local maxima of the matrix are searched by sliding a $w \times w$ grid size window. In our study, we choose $w = 5$. Since building facades, cars and other large objects do not generate local maxima in the $V(x, y)$ matrix, we do not have a challenge to do further processing to eliminate them. However, we cannot prevent false detection of pole like objects. The detected (x_t, y_t) local maximums are assumed as approximate tree trunk positions which unfortunately contains some light poles and traffic signs as well.

2.3 Assigning Points to Two Different Classes

The detected (x_t, y_t) locations are used in the classification process that we perform with the algorithm shown in pseudocode in Figure 3.

In this pseudocode, the $dist$ variable is the minimum $2D$ distance to the closest tree trunk position in the grid. $dist_{thresh}$ is the

```
Line1   for (i=0; i < P_num; i++) {
Line2       X = round(P(i).x –X_min – 0.5*x_step / x_step);
Line3       Y = round(P(i).y – Y_min – 0.5*y_step / y_step);
Line4       Z = P(i).z – Z_min;
Line5       V(X,Y) = V(X,Y) + Z;
Line6   }  //End of the for loop
```

Figure 2: Pseudocode of generating the 2D probability matrix

```
Line1   for (i=0; i < P_num; i++) {
Line2       X = round(P(i).x – X_min– 0.5*x_step / x_step);
Line3       Y = round(P(i).y – Y_min – 0.5*y_step / y_step);
Line4       dist = min(sqrt((X-x_t)^2 + (Y-y_t)^2));
Line5       if (dist < dist_thresh){
Line6               P(i).class = 1;
Line7       }
Line8       else {
Line9               P(i).class = 0;
Line10      }
Line11  }  //End of the for loop
```

Figure 3: Pseudocode of the algorithm for assigning tree class labels to the points

maximum allowed distance to the closest tree trunk position. So, for each 3D point the 2D Euclidean distance to the closest trunk location is determined. If this distance is below a given threshold distance then the point is assigned to that tree trunk.

2.4 Filtering the Ground Points

After assigning "1" and "0" labels to the points (which indicate that they are from the 'tree' class or 'non-tree' class respectively), we apply a filtering process in order to move some of the ground points from the 'tree' class to the 'non-tree' class. To do so, we use a $(2 * dist_{thresh} \times 2 * dist_{thresh})$ size window which is located at the center of each tree trunk. Filtering is applied to the points which fall into this $2D$ grid space limited by the window when they are projected on the (x, y) plane. Inside of the window, each point which has class label "1" is checked, and the point with the lowest z position is selected as a reference (called z_l). Again inside of the window, each point with class label "1" is checked and the points which have z value lower than $z_l + z_{tol}$ is moved to the 'non-tree' class, by changing their class labels to "0". Here, z_{tol} is the tolerance value that we select as 10% of z_l in our application. z_l is computed again for each window position, since it is not a global value for all input point clouds. In Figure 4, we provide the detailed mathematical steps of the ground filtering function.

```
Line1   for (j=0; j < t_num; j++) {
Line2       index = min(round(P.x – X_min – 0.5*x_step / x_step) – x_t) < dist_thresh;
Line3               && min(round(P.y – Y_min – 0.5*y_step / y_step)-y_t) < dist_thresh;
Line4       z_l = min(P(index).z);
Line5       for (k=0; k < size_index; k++) {
Line6               if (P(index(k)).z < (z_l + 0.01*z_l)) {
Line7                       P(index(k)).class = 0;
Line8               }  //End of the if
Line9       }  //End of the for loop
Line10  }  //End of the for loop
```

Figure 4: Pseudocode of the algorithm for filtering the ground points of the tree class.

2.5 Known Issues

Unfortunately this method cannot avoid assigning wrong classification labels to the points which are coming from other objects around the tree such as traffic signs, light poles or people standing very close to the tree trunk. Figure 5 shows a focused view of the tree classification result on a subsection of an MLS point cloud. As it is seen in the bottom figure, the light poles and the traffic lights are labelled as trees in the result point cloud.

Another important issue to take care of is how to choose the z_{tol} value. If it is too high then some lower parts of the tree trunk will be eliminated. On the other hand, however, if the tree is standing on a slope or if the data is noisy (having some redundant very low values) then some of the ground points will not be filtered. In the following section, we provide results of the tree classification algorithm on mobile and airborne laser scanning point clouds.

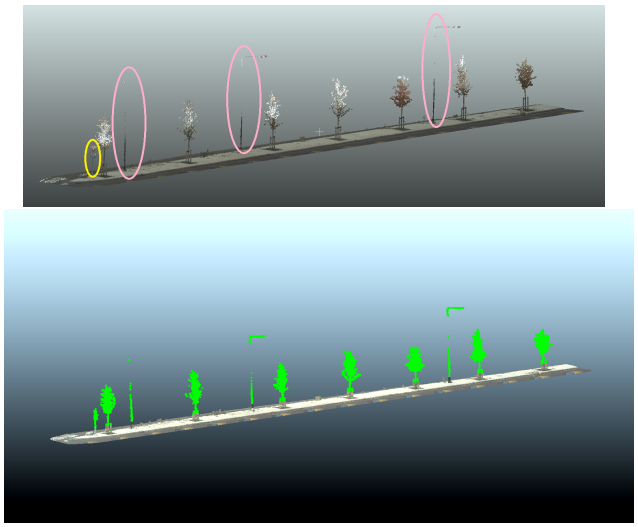

Figure 5: Top; a sub-section of the MLS point cloud having trees, light poles and traffic signs in the same row (light poles and traffic signs are shown with circles). Bottom; tree classification result of the same sub-section, which also holds the false detection of light poles and traffic signs (tree class is indicated by green color).

3 EXPERIMENTS

3.1 Data Description

We test the algorithm on TUDelft campus point clouds which are acquired by two different sensors; one is from a mobile laser scanner and the other one is from an airborne laser scanner. The mobile laser scanning point cloud is acquired by the FUGRO company in 2013. This mobile laser scanner can collect approximately 11500 points per square meter with a ranging accuracy of less than 2 cm. The airborne point cloud is obtained from the free data base called "Actueel Hooghtebestand Nederland" AHN (2008). AHN2 (the second and the higher resolution point cloud data base) data of the study region have been acquired in 2008. The AHN2 data that we use in this study is gridded in (x, y) space at 0.5 meter spatial resolution. In Figure 6, we show the original MLS and AHN2 point clouds together.

In order to evaluate the algorithm performance on different sensors, we take the intersecting part of the point clouds. The upper row image in Figure 8 show the intersection area of the AHN2 and MLS point clouds.

3.2 Classification Results

We test the algorithm on these two point clouds. The bottom row of the same figure shows the tree classification results where points from the 'tree' class are shown with a vibrant green color. In order to be able to talk about the performance of the results we use groundtruth point clouds which we have generated by selecting tree points manually. Table 1 tabulates the performance values for AHN2 and MLS point clouds separately. The column called *'Points'* holds the total number of points in the input point cloud that we process. The *'Tree Points'* column holds the tree points in the groundtruth point cloud. The *'Detected'* column shows the true detection. However the light poles and traffic signs which are very close to the groundtruth points are also labelled as true detection. More detailed manual calculation is necessary when points coming from these non-tree objects are wanted to be eliminated. Unfortunately, the AHN2 data misses small trees since their point density does not appear high enough to be detected. On the other hand, the MLS data misses the trees which are further away from the road where the laser scanning vehicle has travelled. In this case, the trees which are far away from the scanning position appear in lower point densities. In Figure 7, we illustrate the point cloud density of the MLS point cloud by false coloring. The color palette assigns red for the highest and blue for the lowest local point density values. The red color points make the driving trajectory visible and it is seen that the point density decrease when the objects are further away from the laser scanning vehicle.

The last column of the table 1 shows the very low and very promising computation time needed for classifying these large point clouds. The computation time necessary for processing these two point clouds are very similar since the number of points and the surface area size are almost the same. The number of points in the input point cloud directly affects the time needed for probability matrix generation and also for assigning points to two different classes. The ground point filtering step is mostly affected by the grid size selection and the sliding window size. Smaller grid size and larger sliding window increase the computation time. The scene is also changes the computation time needed to process the point cloud. Having more trees in the scene causes having more local maxima and more distance computation in order to assign every point into the correct tree trunk.

Data	Points	Tree Points	Detected	% **TP**	time (sec)
AHN2	231177	120155	111301	**92,63**	29,66
MLS	281405	160807	157225	**97,77**	29,57

Table 1: Tree classification performances on AHN2 and MLS point clouds of a large area at TUDelft campus.

3.3 Case Study

In order to present the classification performances a little bit more clearly, in Figure 9, we present the classification for a small subsection of the MLS point cloud. We also provide Table 2 to give the exact point numbers of the clouds which are presented in the sub-figures.

Figure 9(a) shows the original MLS point cloud section that we consider in more detail. Figure 9(b) shows the point cloud that we have manually labelled for creating a benchmark. The green points belong to the benchmark tree points which are used for classification performance calculation. Figure 9(c) shows the subsampled and automatically classified point cloud. Here, red points represent the points from the 'tree' class and blue points represent the points from the 'non-tree' class. Figure 9(d) shows the same

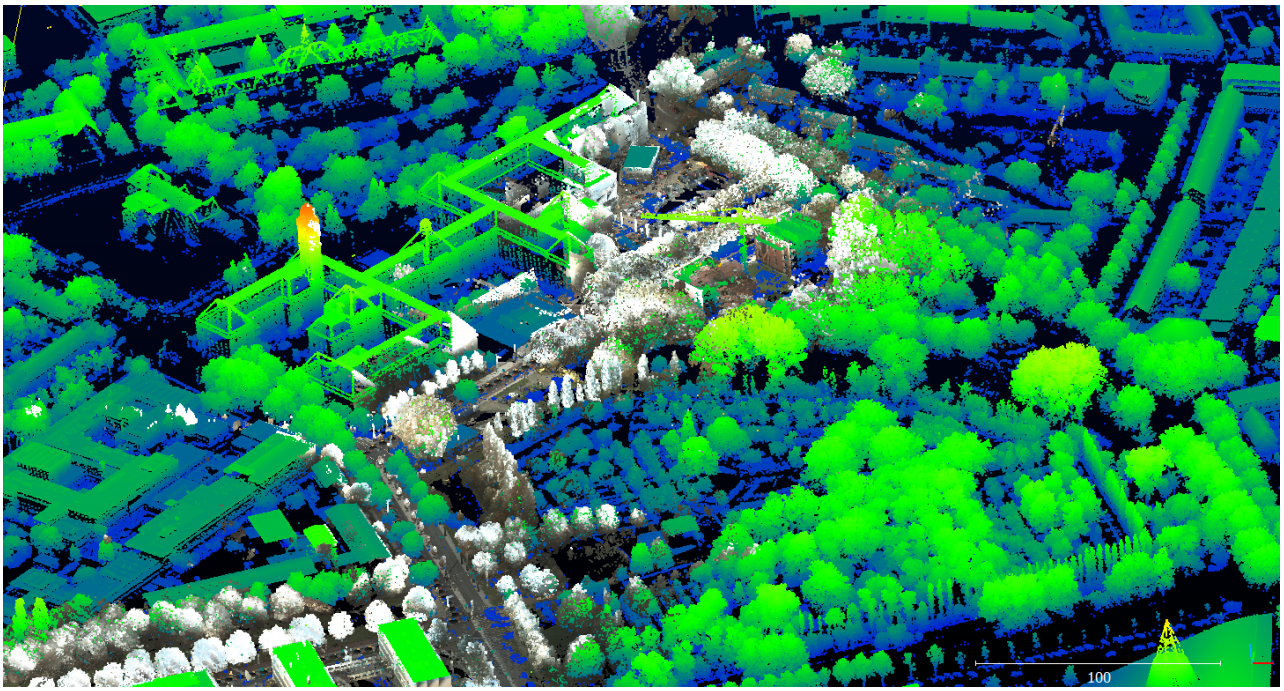

Figure 6: The original AHN2 point cloud (the larger cloud) and MLS point cloud (containing the color information for each point) are shown together.

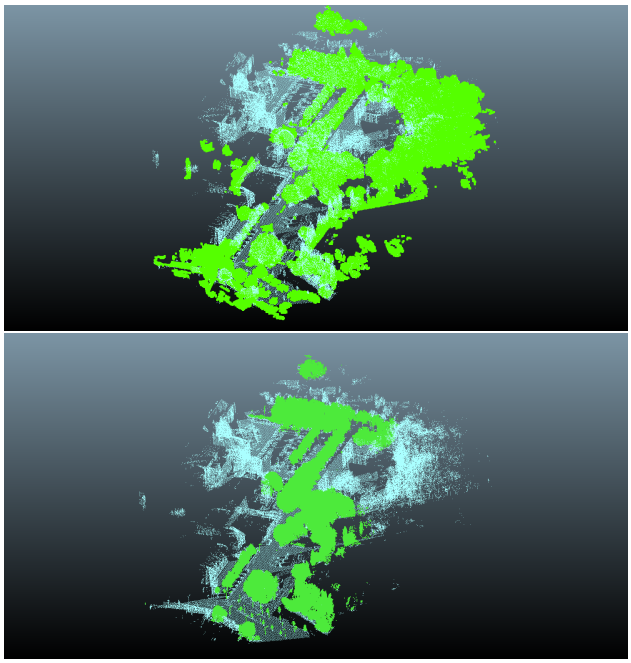

Figure 8: The intersecting AHN2 and MLS points are shown in the top and the bottom row respectively. The points which are classified as trees are shown in green.

result together with the original MLS point cloud. Finally, Figure 9(e) shows the benchmark points (in white color) and the automatically classified point cloud together. Here, blue points are accepted as true detection and the warmer colors are accepted as false detection since they are further away from the benchmark cloud. As can be seen in Table 2, the original MLS cloud has 524,475 points. In order to reduce the computational difficulty, this cloud is subsampled before classification. The subsampled cloud contains 21,485 points and 17,794 points are classified into the 'tree' class automatically. By comparing the 3D Euclidean distances to the closest benchmark points, we have concluded that 202 of 17,794 points are more than $1m.$ further away from the benchmark. We have assumed those points as false detections. We have computed the false detection percentage in this small area as 1.13% and correct detections as 98.86%. As can be seen in the results given in Figure 9, the false detections are coming from the points of light poles and the cars which are parked very close to the tree trunks.

Data	Points
Tree points in the original MLS section (Fig. 9(a), Fig. 9(d))	525,475
Total points in the subsampled MLS section (Fig. 9(c))	21,485
Detected tree points in the subsampled MLS section (Fig. 9(c), Fig. 9(d))	17,794
False tree detection in the subsampled MLS section (Fig. 9(d))	202

Table 2: Tree classification performances on a small section of the MLS point cloud.

4 CONCLUSIONS

In this article, we have introduced a new method for classifying trees from laser scanning point clouds. The reliability of the method is very high by the fact that it requires only few parameters and that results are not heavily affected by slight changes of the parameter values. We test the algorithm on mobile laser scanning and airborne laser scanning data of a region inside of the TUDelft campus. The algorithm distinguishes trees from other urban structures easily without needing to check detailed geometry of the segments. This is the main advantage of the algorithm. The simple yet reliable approach makes the algorithm very fast and gives advantage of running the program on different platforms easily, without having need of heavy mathematical computation methods and library dependencies. Unfortunately, we cannot prevent false detection of light poles and traffic signs in the 'tree' class. In future studies, we will be focusing on providing

mathematical solutions to overcome this challenge. Our current experimental results indicate the reliability of the proposed algorithm and its possible usage when fast and big data processing is needed.

ACKNOWLEDGEMENTS

This research is funded by the FP7 project IQmulus (FP7-ICT-2011-318787) a high volume fusion and analysis platform for geospatial point clouds, coverages and volumetric data set. We also give thanks to the FUGRO company in the Netherlands for providing us mobile laser scanning point clouds acquired from a part of the TUDelft campus area.

References

AHN, 2008. Actual height model of The Netherlands. http://www.ahn.nl/english.php.

Bremer, M., Wichmann, V. and Rutzinger, M., 2013. Eigenvalue and graph-based object extraction from mobile laser scanning point clouds. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences 2(5), pp. 55–60.

Bucksch, A., Lindenbergh, R., Menenti, M. and Raman, M., 2009. Skeleton-based botanic tree diameter estimation from dense lidar data. In Proceedings of SPIE Optics and Photonics, August 2-6, San Diego, US.

Gorte, B. G. H. and Pfeifer, N., 2004. Structuring laser scanned trees using 3d mathematical morphology. ISPRS International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 35, pp. 929–933.

Kaartinen, H., Hyypp, J., Yu, X., Vastaranta, M., Hyyppa, H., Kukko, A., Holopainen, M., Heipke, C., Hirschmugl, M., Morsdorf, F., Naesset, E., Pitkanen, J., Popescu, S., Solberg, S., Wolf, B. and Wu, J., 2012. An international comparison of individual tree detection and extraction using airborne laser scanning. Remote Sensing 4(4), pp. 950–974.

Lalonde, J., Vandapel, N. and Hebert, M., 2006a. Automatic three-dimensional point cloud processing for forest inventory. Technical Report CMU-RI-TR-06-21, Robotics Institute, Carnegie Mellon University.

Lalonde, J., Vandapel, N., Huber, D. and Hebert, M., 2006b. Natural terrain classification using three-dimensional ladar data for ground robot mobility. Journal of Field Robotics 23(10), pp. 839–861.

Li, W., Guo, Q., Jakubowski, M. and Kelly, M., 2012. A new method for segmenting individual trees from the lidar point cloud. Photogrammetric Engineering and Remote Sensing 78, pp. 75–84.

McDaniel, M., Nishihata, T., Brooks, C., Salesses, P. and Iagnemma, K., 2012. Terrain classification and identification of tree stems using ground-based lidar. Journal of Field Robotics 29(6), pp. 891–910.

Monnier, F., Vallet, B. and Soheilian, B., 2012. Trees detection from laser point clouds acquired in dense urban areas by a mobile mapping system. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences 1(3), pp. 245–250.

Pu, S., Rutzinger, M., Vosselman, G. and Oude Elberink, S., 2011. Recognizing basic structures from mobile laser scanning data for road inventory studies. ISPRS Journal of Photogrammetry and Remote Sensing 66(6), pp. 28–39.

Pyysalo, U. and Hyyppa, H., 2002. Reconstructing tree crowns from laser scanner data for feature extraction. International Archives of Photogrammetry and Remote Sensing 34(3B), pp. 218–221.

Raumonen, P., Kaasalainen, S., Kaasalainen, M. and Kaartinen, H., 2011. Approximation of volume and branch size distribution of trees from laser scanner data. ISPRS International Archieves of the Photogrammetry, Remote Sensing and Spatial Information Sciences 3812, pp. 79–84.

Rutzinger, M., Pratihast, A., Oude Elberink, S. and Vosselman, G., 2010. Detection and modelling of 3d trees from mobile laser scanning data. ISPRS Archieves of Photogrammetry, Remote Sensing and Spatial Information Sciences 38(5), pp. 521–525.

Tittmann, P., Shafii, S., Hartsough, B. and Hamann, B., 2011. Tree detection and delineation from lidar point clouds using ransac. Proceedings of SilviLaser 2011.

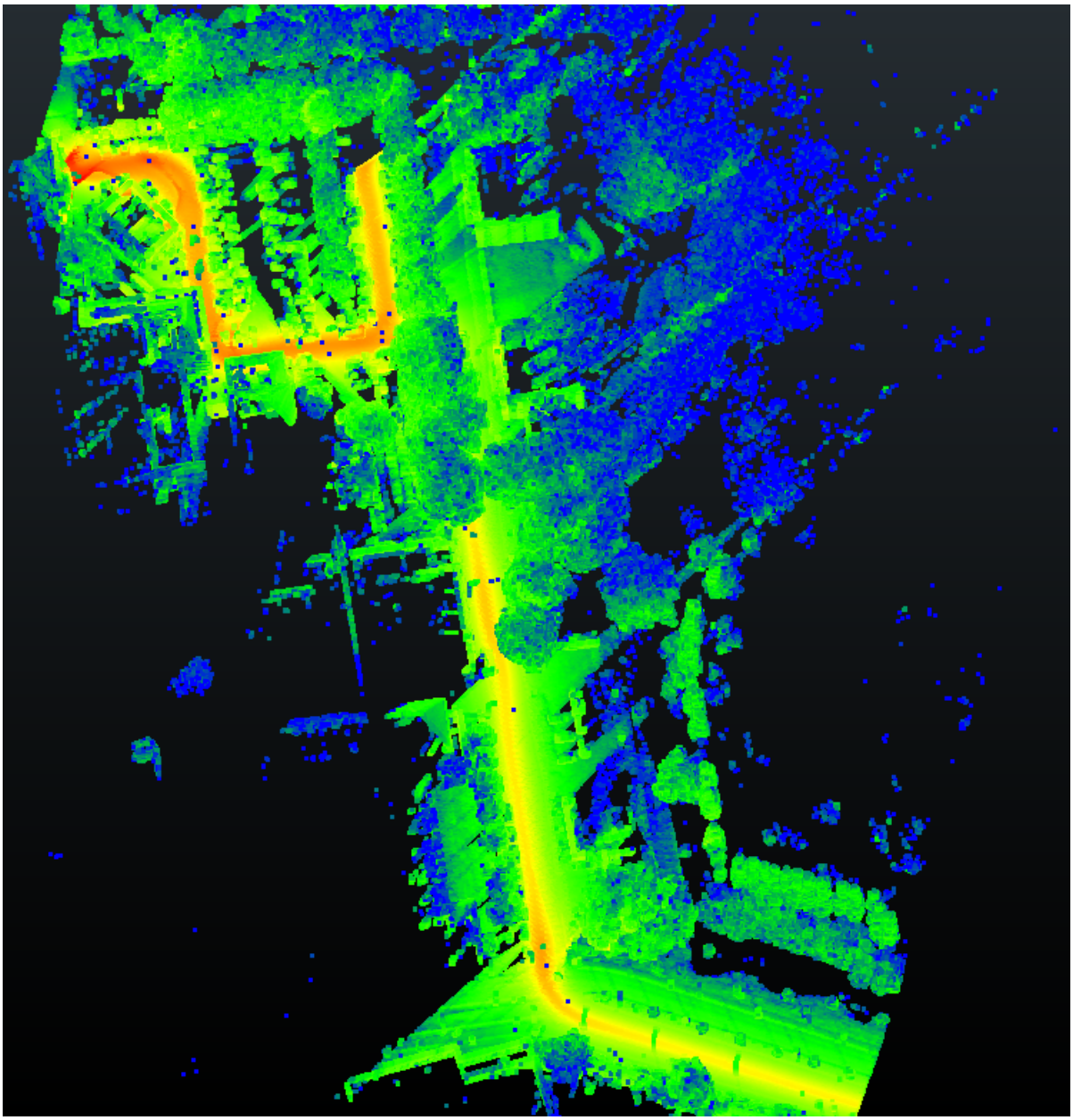

Figure 7: The local point density of the MLS point cloud is illustrated with false color. The warm colors which correspond to high density values also give clue about the driving trajectory of the mobile laser scanning vehicle.

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
Editors: S. Oude Elberink, A. Velizhev, R. Lindenbergh, S. Kaasalainen, and F. Pirotti
doi:10.5194/isprsannals-II-3-W5-137-2015

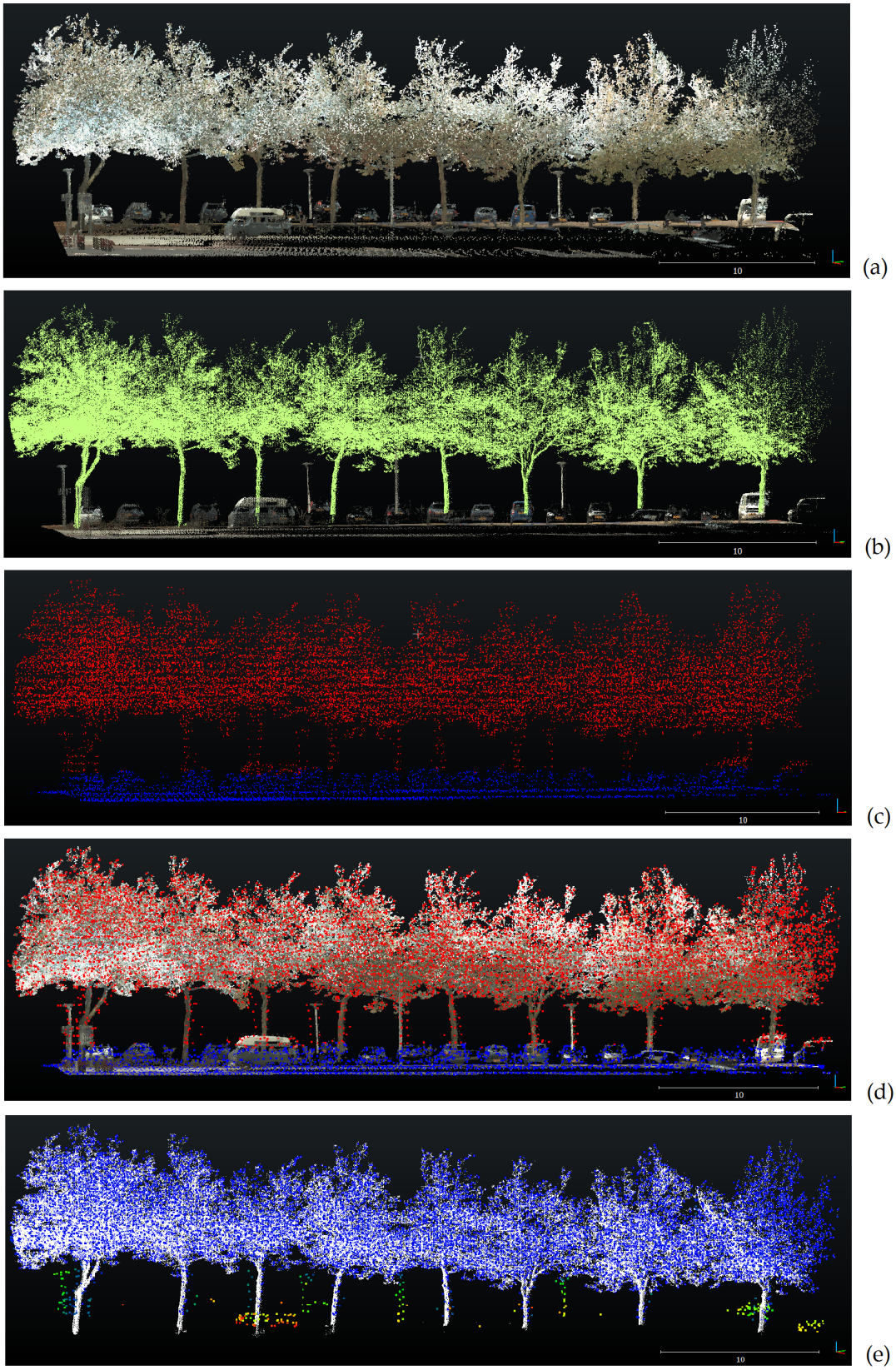

Figure 9: The classification performance on a small section of the MLS point cloud is represented with details. (a) The MLS point cloud section, (b) The manually labelled tree points in the MLS point cloud section (green points are chosen as benchmark for performance calculation), (c) The subsampled and automatically classified point cloud (red points are from the 'tree' class and blue points are from the 'non-tree' class), (d) The subsampled and automatically classified point cloud is shown together with the original MLS point cloud, (e) The benchmark points (in white color) and the automatically classified points are shown together (blue points are accepted as true detection, the warmer colors are accepted as false detection since they are further away from the benchmark cloud).