



Classification and Change Detection in Mobile Mapping LiDAR Point Clouds

Mirjana Voelsen¹  · Julia Schachtschneider² · Claus Brenner²

Received: 23 October 2019 / Accepted: 27 March 2021 / Published online: 6 May 2021
© The Author(s) 2021

Abstract

Creating 3D models of the static environment is an important task for the advancement of driver assistance systems and autonomous driving. In this work, a static reference map is created from a Mobile Mapping “light detection and ranging” (LiDAR) dataset. The data was obtained in 14 measurement runs from March to October 2017 in Hannover and consists in total of about 15 billion points. The point cloud data are first segmented by region growing and then processed by a random forest classification, which divides the segments into the five static classes (“facade”, “pole”, “fence”, “traffic sign”, and “vegetation”) and three dynamic classes (“vehicle”, “bicycle”, “person”) with an overall accuracy of 94%. All static objects are entered into a voxel grid, to compare different measurement epochs directly. In the next step, the classified voxels are combined with the result of a visibility analysis. Therefore, we use a ray tracing algorithm to detect traversed voxels and differentiate between empty space and occlusion. Each voxel is classified as suitable for the static reference map or not by its object class and its occupation state during different epochs. Thereby, we avoid to eliminate static voxels which were occluded in some of the measurement runs (e.g. parts of a building occluded by a tree). However, segments that are only temporarily present and connected to static objects, such as scaffolds or awnings on buildings, are not included in the reference map. Overall, the combination of the classification with the subsequent entry of the classes into a voxel grid provides good and useful results that can be updated by including new measurement data.

Keywords LiDAR · 3D point cloud · Mobile mapping · Change detection · Classification · Segmentation

Zusammenfassung

Klassifizierung und Veränderungsanalyse auf Basis von LiDAR-Punktwolken des mobilen Mappings. 3D-Modelle der statischen Umgebung zu erstellen ist eine wichtige Aufgabe für das Voranbringen von Fahrerassistenzsystemen und dem autonomen Fahren. In dieser Arbeit wird eine statische Referenzkarte aus einem Mobile Mapping “light detection and ranging” (LiDAR) Datensatz erstellt. Die Daten wurden in 14 Messfahrten von März bis Oktober 2017 erhoben und umfassen insgesamt rund 15 Milliarden Punkte. Die Punktwolken werden zunächst mittels Region Growing segmentiert und anschließend mittels eines Random Forest klassifiziert, der die Segmente mit einer Gesamtgenauigkeit von 94% in fünf statische Klassen (“Fassade”, “Pfahl”, “Zaun”, “Verkehrszeichen” und “Vegetation”) und drei dynamische Klassen (“Fahrzeug”, “Fahrrad”, “Person”), einteilt. Alle statischen Objekte werden in ein Voxel-Gitter eingetragen, um verschiedene Messepochen direkt miteinander vergleichen zu können. Im nächsten Schritt werden die klassifizierten Voxel mit dem Ergebnis einer Sichtbarkeitsanalyse kombiniert. Dafür wird eine Strahlverfolgung (Ray Tracing) durchgeführt, um vom Laserstrahl durchquerte Voxel zu erkennen und zwischen leerem Raum und Verdeckung unterscheiden zu können. Jedes Voxel wird anhand seiner Objektklasse und dem jeweiligen Belegungsstatus der verschiedenen Epochen als für die statische Referenzkarte geeignet eingestuft oder nicht. So wird vermieden, statische Voxel, die in einigen Messepochen verdeckt wurden, als

✉ Mirjana Voelsen
voelsen@ipi.uni-hannover.de

Julia Schachtschneider
julia.schachtschneider@ikg.uni-hannover.de

Claus Brenner
claus.brenner@ikg.uni-hannover.de

¹ Institut für Photogrammetrie und GeoInformation, Leibniz
Universität Hannover, Hanover, Germany

² Institut für Kartographie und Geoinformatik, Leibniz
Universität Hannover, Hanover, Germany

ungeeignet einzustufen (beispielsweise Teile eines Gebäudes, die von Bäumen verdeckt wurden). Segmente, die jedoch nur temporär vorhanden aber mit statischen Objekten verbunden sind, wie beispielsweise Gerüste oder Markisen an Gebäuden, sind nicht in der Referenzkarte enthalten. Insgesamt liefert die Kombination der Klassifizierung mit dem anschließenden Eintragen der Klassen in ein Voxel Gitter gute und brauchbare Ergebnisse, die sich durch den Einbezug neu aufgenommener Messdaten aktualisieren lassen.

1 Introduction

Driver assistance systems and functions of semi-automated driving are already an integral part of new vehicles and help to relieve the driver to increase road safety. Research is tending more and more towards (partially) autonomous vehicles that can automatically recognise obstacles and react accordingly. 52% of the worldwide patents in the field of automated driving come from Germany and the development potential is still enormous and wide-ranging (Eckstein et al. 2018).

When using these systems, it is essential that they have a highly accurate and up-to-date 3D model of the environment that can integrate changes. The first step to create such a model is to record the environment. One approach for this is Mobile Mapping, in which the environment is measured, e.g. by mobile laser scanners, cameras or a combination of both. In addition, these systems are usually equipped with a GNSS system. During the next step of data processing, it is important to distinguish between temporal and static objects. Therefore, the classification of single points or segments is useful. As a result the individual elements are each assigned to a class. Another approach is change detection. Here, several measurement epochs are compared to determine whether objects have changed or not. On the one hand this can be done only for static objects, for example to detect demolitions or new buildings. Thus, it is possible to update the environment model with new measurement data. On the other hand, dynamic objects can also be analysed, e.g. to detect parking lots or areas which are often crowded with pedestrians.

In this study an environment model is created by the classification and detection of changes in Mobile Mapping LiDAR point clouds. The data processing uses the following steps: (i) segmentation of the point clouds using region growing, (ii) setting up a feature vector for the segments, (iii) classification of the segments using a random forest, (iv) insertion of the classes into a voxel grid with an edge length of 10 cm, and (v) comparison of the classes and occupancy state of the voxels for the different measurement epochs.

2 Related Work

Mobile Mapping is used for a wide range of applications. It is important that the data acquisition and processing are well adjusted to the desired result. In this section we summarise

related work on point cloud classification (Sect. 2.1), followed by applications of change detection (Sect. 2.2). Related datasets are described in the next (Sect. 3).

2.1 Classification of LiDAR Point Clouds

To classify point clouds with traditional classifiers such as random forest, typically geometric features based on the 3D points within a defined neighbourhood are extracted in a first step (Weinmann et al. 2015). These features are later used for the classification of the point cloud. The underlying assumption is that neighbouring pixels tend to be correlated, which can be valuable additional information in the classification process. As we will use context information from the neighbouring pixels in our method the focus in this section is on context-based algorithms.

Weinmann et al. (2015) compare different kinds of neighbourhoods, geometric features, approaches for feature selection and classifiers. They conclude that an individual neighbourhood selection, based on their method of eigenentropy-based scale selection, significantly improves the classification result. They also use the random forest classifier as a good trade-off between classification accuracy and efficiency. Landrieu et al. (2017) construct a regularisation framework based on structured optimisation to smooth semantic labelling from any classification result of a 3D point cloud. They improve the results after experimenting with different regularisers and fidelity functions.

Another approach to include context information is to use object segmentation as a preprocessing step. Afterwards the individual objects are classified and all points of the object are assigned to the same class. A disadvantage of this method is that the accuracy of the classification is highly dependent on the segmentation. Therefore, it is important to use a well-performing segmentation algorithm. Grilli et al. (2017) give an overview of different point cloud segmentation algorithms and their strengths and weaknesses.

It is important to find appropriate features to describe the individual segments adequately. Lehtomäki et al. (2016) analyse and compare three different groups of object features: local descriptor histograms (Himmelsbach et al. 2009), spin images (Johnson and Hebert 1999) and features that describe the general shape and point distribution of an object. Before classification, the ground and buildings are removed and the remaining point cloud is segmented. Using support vector machines (SVM) the segments are classified into “tree”,

“street lamp”, “traffic sign”, “vehicle”, “pedestrian” and “object groupings”. The additional use of the new feature groups improved the classification accuracy by 9.8% compared to the classification where only the features for general shape and point cloud distribution were used.

During the past years, neural networks have been established as the new, state-of-the-art method for several classification tasks, including semantic segmentation of images and point clouds. Qi et al. (2017) achieve state-of-the-art results using their PointNet architecture which can directly operate on point clouds and can be used for tasks like object classification and part segmentation. This network was extended for example by Engelmann et al. (2017) who incorporated a larger receptive field to further improve the results. Other deep learning approaches like the one proposed by Landrieu and Simonovsky (2018) consider the spatial relations between neighbouring points using a structure called superpoint graph that offers a rich representation of contextual relationships. As most deep learning models need a large amount of training data, Hackel et al. (2017) presented a new 3D point cloud classification benchmark dataset with more than four billion manually labelled points.

2.2 Change Detection

Change detection is used in most cases to identify static objects in point clouds. Lindenbergh and Pietrzyk (2015) and Qin et al. (2016) present reviews of different methods for change detection and deformation analysis. While many approaches deal with airborne data and use digital surface models (DSMs), there are only a few using point clouds of urban areas, acquired by Mobile Mapping Systems. Related to our work, the approaches for change detection can be divided into three categories: point/ ray based, voxel based and object based.

Hebel et al. (2013) align the actual measurement to a reference and then perform a point-wise analysis of the point cloud. In order to deal with penetrable objects and occlusion, they first apply a pre-classification of vegetation. Then they use ray tracing to determine the occupancy state of the free space: the space along the laser ray is empty and the space behind the reflecting point is unknown. These states can be updated with each laser ray that traverses space. The approach of Xiao et al. (2015) applies a similar model to Mobile Mapping LiDAR point clouds. Their model is also based on the scanning rays and the local point distribution. It directly evaluates the consistency of points. In contrast to Hebel et al. (2013) they do not use a pre-classification and combine their method with a distance-based approach to avoid conflicts with permeable objects.

Gehring et al. (2017) work on multi-temporal Mobile Mapping LiDAR point clouds. They use an octree for their map representation and perform a plane-filtered ray casting

to determine the voxel occupancy states (“occupied”, “moving object” or “residual”). This way they are not only able to filter the environment for static objects, but also remove artefacts caused by discretization, especially near to planar structures. The work of Fuse and Yokozawa (2017) focuses on detecting changes with low-cost sensors. Therefore, they use Mobile Mapping data to generate artificial measurements of realistic traffic participants, which serve as input for the actual simulation. They use an occupancy grid to model the environment and use Dempster–Shafer theory to distinguish between occlusion and real changes.

Aijazi et al. (2013) use an object based approach to detect and analyse changes in an urban environment. They use multi-temporal Mobile Mapping data. In the first step, they segment the point clouds of the individual measurement runs and perform a classification into the two classes “permanent” and “temporal” using geometrical models and local descriptors. All temporal and unclassified objects are removed and the remaining point clouds are aligned by an ICP (iterative closest point) algorithm. The result is mapped into 3D evidence grids. Schachtschneider et al. (2017) combine an occupancy grid with an object-based approach. Therefore, LiDAR point clouds of several epochs are first registered and segmented into objects. Then an occupancy grid is created by tracing each individual laser ray to determine the behaviour of each object. For each voxel it is stored whether it is “free”, “occupied” or “not seen”. Objects are classified as static or dynamic using thresholds, e.g. an object is static if at least 25% of all voxels in that segment are occupied in at least 70% of all measurement runs. Thus, temporary objects can be removed from the occupancy grid and only static objects remain, even if they have been partially occluded. In this paper, we use the results of an object classification together with a visibility analysis of the voxels to determine their probability to belong to a static object. Therefore, we first perform an object segmentation (Sect. 4.1) on the point clouds of the individual measurement runs. For each object, we calculate a set of feature values. Then the objects are classified using a random forest (Sect. 4.2). The classification result is inserted into a voxel grid. This way the different measurements can be compared and filtered to create the static map. In addition, data from a visibility analysis are used to detect occlusions (as in Schachtschneider et al. (2017)). This two-step approach has several advantages: The object-wise classification has the benefit that objects are labelled as static in their full extent. There are no outlier points, e.g. at edges or in areas that are only rarely measured. Furthermore, the classification makes it possible to distinguish objects for which the voxel occupancy can be very similar over time, e.g. vegetation and vehicles on frequently occupied parking lots. By combining the classification with the visibility analysis, we are able to filter temporary parts like construction scaffolds or extended sunblinds which

could not be separated by the region growing segmentation from static objects. The result is a static reference map of an urban environment.

3 Data

The data for this project were collected using a Riegl VMX-250 Mobile Mapping System installed on the roof of a measurement van (see Fig. 1). It consists of two laser scanners, a camera system and a localisation unit. Each of the laser scanners acquires 100 scan lines, and up to 300,000 measured 3D points, per second. The measuring range is between 1.5 and 200 m. For a distance of 50 m, the accuracy is specified as ten millimetres, with a precision of five millimetres. The localisation unit consists of a highly accurate GNSS/IMU combined with a Distance Measurement Instrument (rotary encoder, attached to one wheel). The accuracy of the trajectory is 10–30 cm in height and 20 cm in position, under good conditions. Further specifications of the Mobile Mapping System can be found in RIEGL Laser Measurement Systems (2012).

The data used in this work were collected by a 1-year measurement campaign along a 20-km route in Hannover, Germany (see Fig. 2). The whole measurement campaign consists of 26 measurement runs which were performed about every other week. Figure 3 shows that the dataset covers all different seasons and various lighting and weather conditions.

In Table 3 in the appendix, we summarised a number of freely available LiDAR datasets. We note that our dataset features a special combination of characteristics:

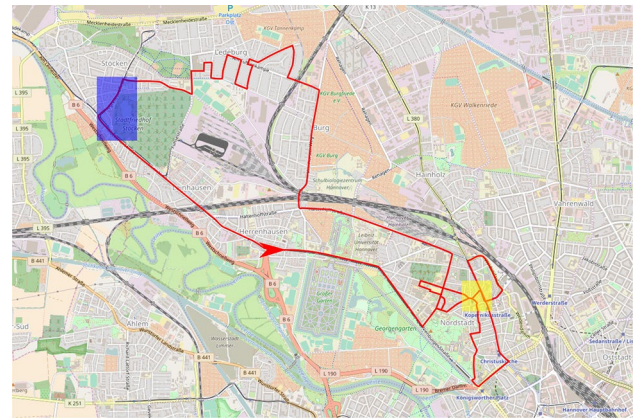


Fig. 2 Measurement area, red: route, yellow: bounding box “Nordstadt”, blue: bounding box “Stöcken”, background map: OpenStreet-Map Foundation (2019)

- Long-term measurements / repeated measurements of the same area (one year of biweekly measurements, different seasons, weather and lighting conditions).
- Large measurement area with a variety of different scenes (20 km route with inner city streets, residential areas, tram lines, various intersections, areas with high pedestrian and bike traffic, parking lots).
- High point density, very high accuracy LiDAR measurements and very precise alignment across measurement campaigns.



Fig. 1 Measurement van with Riegl VMX-250



Fig. 3 Example images of the same scene in different seasons **a** 17-03-28, **b** 17-04-28, **c** 17-06-06, **d** 17-06-20, **e** 17-08-08, **f** 17-10-04, **g** 17-11-07, **h** 18-01-17, **i** 18-02-15

Most other datasets only contain one or very few measurement runs of the same area. Only the NCLT dataset (Carlevaris-Bianco et al. 2016) and the Oxford dataset (Maddern et al. 2016) are long-term datasets with multiple runs during different seasons. The NCLT dataset was acquired using a Segway robot. The same area (about a square kilometre around the University of Michigan North Campus, US) was surveyed with several varying routes (about 5.5 km each, indoors and outdoors) over a period of 15 months in approximately biweekly measurements. The Oxford dataset was acquired using the Oxford RobotCar platform, driving a route in central Oxford, UK. The measurements were performed twice a week for more than 1 year. Both datasets cover a smaller area than ours and have lower point density due to the sensors used. Other datasets provide annotated LiDAR data. The Semantic3D.net has the largest amount of annotated points, four billion manually labelled points with eight classes. This dataset was acquired by terrestrial laser scanners and contains various different scenes of central Europe. The KITTI dataset is also worth mentioning, as it contains a large number of benchmarks. It was acquired in the metropolitan area of Karlsruhe.

For our work, we used a subset of 14 measurement runs (recorded March–October 2017) of the whole biweekly measurement campaign. Figure 4 shows the temporal distribution of the measurement runs used in this paper. In order to be able to examine the recorded point clouds for changes after a successful classification, data from two partial sections of the whole measuring area are used. The first area is located in the district “Nordstadt” of Hannover (see Fig. 2, yellow box) and consists of 892 million points. The second section is located in the district “Stöcken” and contains 489 million points (see Fig. 2, blue box). The first area is close to the city centre while the second area is a residential area in a suburb and contains more trees and other vegetation. The used laser scanning point clouds are already pre-processed

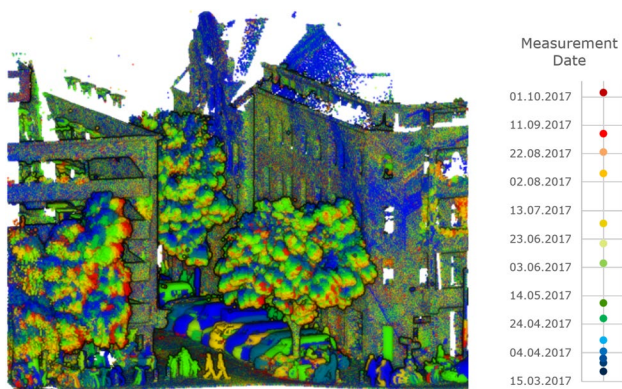


Fig. 4 Example view and temporal distribution of the data. Left: aligned point cloud of 14 campaigns, coloured by campaign. Right: campaign colours and distribution of campaigns over the year

by the Riegl software. Additionally, a bundle adjustment is performed as described in Brenner (2016) to align all measurement campaigns. Afterwards, the precision of the point clouds is usually below one centimetre (Brenner 2016).

4 Methodology

This section describes the individual steps of data processing (see Fig. 5): The pre-processed and aligned point clouds are segmented using region growing (Sect. 4.1) and classified using a random forest, based on features describing the local point geometry (described in more detail in Sect. 4.2). The resulting labelled objects are then inserted into a voxel grid. In the end, the classification results for the different measurements are compared and combined with a temporal analysis for change detection (Sect. 4.3).

4.1 Segmentation

In this work we consider the object-wise classification of urban objects. Therefore, we perform an object segmentation in the first step. The segmentation of the point clouds is done using a region growing, similar to the one used by Schlichting and Brenner (2016a). First, the ground is removed as one segment using the height and normal vectors. This method is suitable for flat areas and needs to be adapted for areas with a slope. Afterwards, all other segments are separated using a distance threshold of 10 cm. Conditions for instantiating a new segment are a minimum number of points ($n_{\min} = 500$) and a minimum height ($h_{\min} = 20$ cm). In this way, segments that are too small are excluded and do not lead to unnecessary misclassifications. The parameters n_{\min}

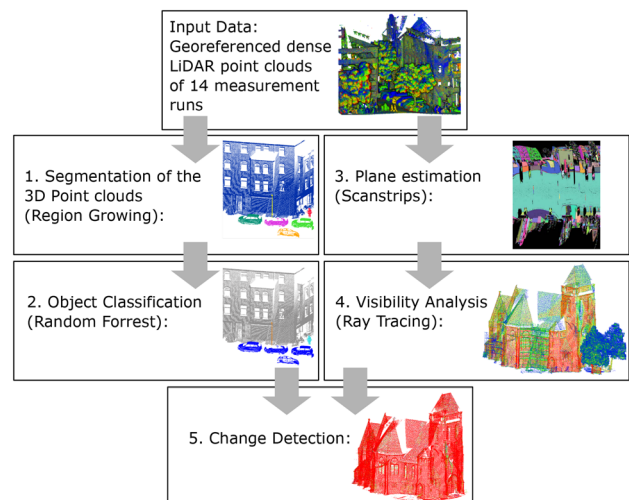


Fig. 5 Overview of the individual steps of the change detection process

and h_{\min} were chosen based on experience. Segments with a smaller n_{\min} and h_{\min} can usually not be recognised by humans and, therefore, cannot be labelled manually. To avoid problems with large amounts of data during the segmentation process, the recorded point clouds are divided into tiles of 15×15 m edge length (in x and y direction) before segmentation. In Fig. 6 some examples for segments of different classes are shown.

4.2 Classification

The segments are classified using a random forest classifier. We used the Cityscapes dataset (Cordts et al. 2016) as a reference and adapted their classes to our data. As a result, we use the following eight classes: “facade”, “vehicle”, “person”, “pole”, “fence”, “traffic sign”, “bicycle” and “vegetation”. During the manual creation of the training data, the class “without class” is additionally used to exclude segments that cannot be assigned to one of the eight classes. Since some objects could not be separated by the region growing with a threshold of 10 cm, e.g. vegetation grown around traffic signs or objects (e.g. bicycles) leaning against facades, it is important that these merged objects are not included in the training data. We also took care that segments from both measurement areas and different time epochs were labelled. This is especially important for vegetation which was still leafless during the first measurement runs. The total amount of training data is 6108 segments.

To train the classifier a total number of 383 features is calculated for each segment as follows:

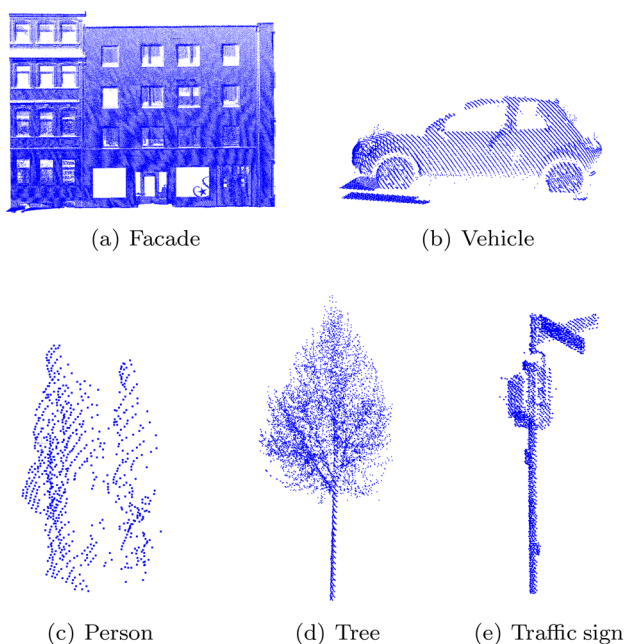


Fig. 6 Examples for segments

- Number of points inside one segment (1 feature).
- Covariance matrix, eigenvalues and eigenvectors of the scatter matrix (6, 3 and 9 features, respectively).
- Axis-parallel bounding box: This box contains all the points of the segment and is aligned parallel to the x , y and z axis (3 features).
- Oriented bounding box: This box also contains all points of the segment but is oriented according to the calculated eigenvectors of the segment (3 features).
- Height above the ground: The difference between the mean height of the corresponding ground segment and the mean height of the segment (1 feature).
- Height difference between laser scanner and segment: Height difference between the laser scanner and the mean height of the segment points (similar to height above the ground, 1 feature).
- Constant cylinder distribution: For cylinder plates with a height of 50 cm the percentage share of points is calculated (40 features for a maximum high of 20 m for objects).
- Dynamic cylinder distribution: For four “dynamic” (adjusted in height) cylinder plates the percentage share of points is calculated. Each of the plates has a height of 1/4 of the total height of the segment (Similar to the calculation in Lehtomäki et al. (2016), four features).
- View point feature histogram (VPFH): With this the surface of the segment is described using the points and normal vectors. See the work of Rusu et al. (2010) for the exact calculation (308 features).
- Linearity, planarity and scatter: describe whether the points inside a segment are distributed rather linear, planar or scattered. The exact calculation is described in Demantke et al. (2011) (3 features).
- Mean distance between each point and the centre of the segment (1 feature).

In order to find the best parameter settings for the random forest classification, we tested possible parameter sets by a grid search and picked the best choice. These parameters include the maximum depth of the decision trees, the minimum number of training data for a leaf, the minimum number of training data for a node, and the total number of decision trees to train. To split the samples assigned to a node in the decision tree, the Gini impurity is used (Breiman 1984). Its value is between zero and one, where a value of zero occurs if all samples belong to the same class. The Gini impurity is calculated for a subset of samples $\mathbf{X} = \{x_1, x_2, \dots, x_J\}$ at node t using equation 1, where J is the number of samples at this node, and p_c is the proportion of samples belonging to class c , $1 \leq c \leq C$, which is computed from the absolute counts n_c using $p_c = n_c/J$:

$$I(p_1, p_2, \dots, p_C) = 1 - \sum_{c=1}^C p_c^2. \tag{1}$$

For each node, a feature and a particular threshold is selected, subdividing \mathbf{X} into two subsets \mathbf{X}_1 and \mathbf{X}_2 . Where the optimal selection is obtained if the weighted sum of their corresponding Gini impurities I_1 and I_2 is smallest, the weights are selected according to the sizes of the two subsets, J_1 and J_2 . For example, for $C = 2$, the largest possible impurity $I(p_1, p_2) = 0.5$ would occur for $p_1 = p_2 = 0.5$, and an optimal split would separate the two classes, each subset would be pure ($I_1 = I_2 = 0$), leading to $J_1/J \cdot I_1 + J_2/J \cdot I_2 = 0$.

As some of these features are redundant, a selection of the most suitable features is included after the training process, to analyze their influence on the classification. For this purpose, the Gini Impurity is used to calculate the Gini importance. At each node of a tree, the goodness-of-split is defined as the difference of the impurity of the parent node and the impurities of the left and right child, weighted by their corresponding proportions of samples (Breiman 1996). This decrease in impurity, caused by the selection of a specific feature for the split, is summed over all nodes, weighted by the proportions of samples reaching the nodes. Thus, for each tree and each feature, a total decrease in node impurity caused by that feature is obtained. Finally, these per-tree importances are averaged over all trees to obtain the overall feature importance.

4.3 Detection of Static Objects Using a Voxel Grid

We use a grid with voxels of 10 cm edge length to store our data for the static reference map. In order to create the voxel grid, we merge all classified segments of all epochs and store one class for every measurement run for every voxel. If more than one class for the same measurement run lies within one voxel the closest one is retained, but this should not happen normally because the distance threshold for the object segmentation is also 10 cm.

The detection of static objects uses two steps: first, the voxels are divided into static and temporary according to their class. The following classes are declared as static: “Facade”, “fence”, “pole”, “traffic sign” and “vegetation”. The class “vegetation” is only semi-static as it behaves differently from the other static classes, for example trees and bushes remain static in one place, but move and grow, lose leaves or are cut depending on the seasons or human intervention. In the second step, misclassifications and objects that only occur temporarily in a few campaigns shall be

removed. For example, a scaffold should not be included in the static map, even if the segment was assigned to a static class (fence or facade). In order to delete these elements from our map, we perform a visibility analysis. Therefore, we use a ray tracing algorithm similar to Schachtschneider et al. (2017). Each voxel (edge length 10 cm) stores an occupancy state (“hit” or “traversed” by the laser ray or “not seen”) for each measurement run as a sequence of observations. Here, the method used in Schachtschneider et al. (2017) was slightly adapted. In a pre-processing step, scan strips are segmented into objects with continuous surfaces as described in Brenner (2016) (see Fig. 5, step 3), which are then used to enter planar surface elements into a voxel grid. Instead of just checking whether a voxel was traversed by the laser ray, now it is checked whether one of the estimated plane elements was traversed. The scan-strip segmentation discards small plane elements. This means, objects with a very rough surface or high curvature (e.g. tree foliage), which consist of many very small plane elements) are removed. The data of the ray tracing are involved to get the additional information whether a voxel was traversed by a laser ray. The combination of the classification and the visibility analysis results in the following four cases that can be distinguished for each measurement run:

- Static class and no traversing ray: normal case when an object is in this voxel.
- Static class and traversing ray: this causes a conflict that must have occurred during data processing. For the further processing it is assumed that the entered class is correct.
- No class and no traversing ray: an occlusion can be assumed if a static object was measured in another epoch.
- No class and traversing ray: the voxel is empty.

In order to detect static objects, we calculate a value W_{static} for each voxel, which is a measure for the voxel being static, as follows:

$$W_{\text{static}} = \frac{k}{\text{epochs} - v}, \tag{2}$$

Where k is the number of times a static class within a voxel is measured during the measurement runs and v is the number of occlusions. Epochs is the total number of measurement runs which is 14 in the case of this study and thus corresponds to the maximum times a voxel can be observed. The value for W_{static} lies between zero and one, where larger values indicate a static object.

5 Results

This section presents the results of the individual processing steps described in Sect. 4. First, Sect. 5.1 deals with the segmentation results. Section 5.2 presents the results of the classification, and Sect. 5.3 compares them with results from other authors. Finally, Sect. 5.4 presents the created reference map.

5.1 Segmentation Results

The segmentation of the ground and all other segments works very well for standard situations. After the ground segmentation, all other objects are no longer connected and can be stored as single segments with a distance threshold of 10 cm. Figure 7 shows the segmentation result of an example scene. If different objects are too close to each other (point distance below 10 cm), they may become merged into one object. For example, in Fig. 7, objects close to the facade are not separated by the segmentation approach. The same may happen, e.g. with street signs close to vegetation or persons with a low distance to each other (see Fig. 6c). Another problem may arise if several ground heights occur within one point cloud. With the used algorithm, only the lowest ground level is extracted and the objects remain connected if more ground levels exist. Consequently, the objects cannot be segmented correctly. An example of such a case is shown in Fig. 8. Here only the lowest ground level within a subway shaft is extracted as ground (shown in red).



Fig. 7 Segmentation result of an example scene

5.2 Classification Results

The best parameter combination for the random forest classifier, detected by the grid search as described in Sect. 4.2, is as follows:

- Maximum depth: 25.
- Minimum number of training data for a leaf: 1.
- Minimum number of training data for a node: 2.
- Number of decision trees: 150.
- Used features: 99.

For the selection of the features we took into account the Gini Importance as described in Sect. 4.2. The 99 features with the highest Gini Importance were used for training. Following is a summary of the most important features:

- Mean height difference between laser scanner and segment and height above the ground: Both features indicate the height of the segment, but with different references. Both features make it easy to distinguish small objects (persons, bicycles) from medium-sized objects (trees) and large objects (facades).
- VPFH: With the help of the calculated surface component, the distribution of the normal vectors of an object is well described, allowing a clear angle histogram per class to be established during training.
- Constant cylinder distribution: Especially the lower cylinder plates (up to a height of approx. 4 m), in which most objects are located, have a large impact on the classification results.
- Oriented bounding box: The oriented bounding box seems to adapt better to the individual objects and is, therefore, more meaningful than the axis-parallel bounding box.

Other features, like the eigenvalues, covariance matrix or linear, planar and scatter features are also used but have a

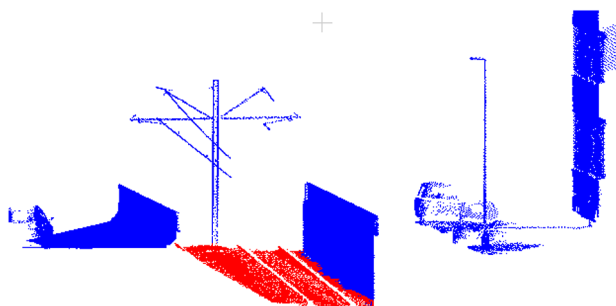


Fig. 8 Segmentation of the ground in the area of the underground shaft (red: subway shaft, detected as “ground”; blue: remaining point cloud)

much lower impact on the classification result. For training, we randomly picked 80% of the labeled segments, testing on the remaining 20%. The classification achieves an overall accuracy of 94% with the test data. The values for Precision, Recall, F-Score and Support are shown in Table 1. The exact calculation of these values can be found in Sokolova and Lapalme (2009).

With a value of 0.71 for the F-score, the class “bicycle” achieves the worst results. This may be due to the fact that this class contains both freestanding bicycles and cyclists and in some cases also multiple bicycles joined into one object. Another reason may be the small amount of training data available for this class (154 training segments). A positive example of a class with little training data is the class “pole”. Here the F-score reaches 0.88 though there are only 140 training segments. The reason for this is that the shape of this class (long, vertical bar) clearly distinguishes it from the other classes and is very consistent within the class.

When analysing the other misclassifications using the confusion matrix in Table 2, it appears that they are quite distributed and do not occur mainly between two or three similar classes. When the classifier is applied to new data,

Table 1 Precision, Recall, F-score and Support of the individual classes with the selected classifier random forest with eight classes

Class	Prec.	Recall	F-Sc.	Support
Bicycle	0.76	0.67	0.71	33
Vehicle	0.93	0.94	0.93	151
Fence	0.89	0.83	0.86	60
Facade	0.94	0.97	0.96	351
Pole	0.95	0.82	0.88	22
Person	0.81	0.96	0.88	45
Traffic sign	0.90	0.90	0.90	82
Vegetation	0.98	0.96	0.97	478
Mean/sum	0.94	0.94	0.94	1222

more challenges arise: Some misclassifications occur due to small segments inside buildings. These fragments, scanned through windows, are difficult to assign to a class. A solution to avoid this would be the exclusion of areas behind facades before the classification. Another problem are merged objects that contain different classes and thus cannot be assigned clearly to one class. One way to further improve the classification would be the introduction of more training data. This is particularly necessary if the classification shall be extended to other regions with different characteristics (e.g. other types of buildings or vegetation).

5.3 Comparison with Classification Results from Other Authors

Since there is no other work available that uses the same data and classes, a direct comparison of the results is not possible. Nevertheless, we want to rank our results in order to evaluate their quality. Therefore, we compare our outcome with similar work.

Lehtomäki et al. (2016) achieve an average overall accuracy of 88.1% using a support vector machine classifier, which is nearly 6% worse than our results. They compare new features, such as Spin Images or Local Descriptor Histograms, to traditional features such as the point distribution of a segment. Since the focus of their work is on new features, they do not fix errors that occur during segmentation, which can have a negative impact on the overall result. The classification divides the segments into the following six classes: “tree”, “lantern”, “car”, “pedestrians”, “traffic signs” and “hoarding”. An exception is the class “construction fence/hoarding” with a Recall value of 0.67. The authors explain this by the fact that this class is partly mixed up with the class “traffic signs”, because both classes contain vertical planes. The class “pole” or “lantern” is classified even better than in our work. This may be due to the fact that a pre-selection of segments is performed before the classification.

Table 2 Confusion matrix of the random forest classification with eight classes

	Predicted class	Actual class							
		A: Bicycle	B: Vehicle	C: Fence	D: Facade	E: Pole	F: Person	G: Traffic sign	H: Vegetation
A	22	0	0	0	0	0	0	0	7
B	4	142	0	3	0	1	0	0	3
C	0	1	50	3	0	0	0	0	2
D	0	6	9	341	0	0	4	3	3
E	0	0	0	0	18	0	1	0	0
F	7	2	0	1	0	43	0	0	0
G	0	0	0	0	4	0	74	4	4
H	0	0	1	3	0	1	3	459	

Thus, in our case, segments which are too small or consist of too few points are filtered out and are not classified.

Paul et al. (2012) divide segmented point clouds into the following six classes using a Gaussian process classifier: “building”, “tree”, “ground”, “hedge”, “vehicle” and “background”. Only the classes “building”, “tree” and “vehicle” can be compared with the results of this paper. The Recall values for the classes “building” and “tree” are close to our results and lie in a very good range from 0.98 to 1.0. The class “vehicle”, with a Recall of 0.64, is classified much worse than in Lehtomäki et al. (2016) and our work. This can be explained by the number of available training data: in Paul et al. (2012) there are 74 training segments of the class “vehicle”, while we use 547 segments. This difference is a clear indication of how important the amount of training data is for the classification.

The comparison has shown that the amount of training data plays a key role in the classification and it should be invested in the production of meaningful training data. In addition, the selection of suitable classes plays an important role, as similarities lead to misclassifications that degrade the overall result.

5.4 Reference Map

The results of the static reference map based on the classification, the voxel grid and the ray tracing can be shown best by an example. For this purpose, in Fig. 9 the church “Lutherkirche” in the “Nordstadt” district of Hannover was chosen. Each voxel is coloured according to its number of occurrences during the 14 measurements. The blue colour indicates rarely occurring voxels (1–3 occurrences, which

corresponds to 7–21%), green slightly more frequent voxels (4–9 occurrences, which corresponds to 28–64%), and red represents very often occurring voxels (12–14 occurrences, which corresponds to 85–100%). It is easy to see that the roof, the scaffolding and also the side wall of the building, which is partially covered by the trees in front of it, were scanned significantly fewer times than other parts of the facade.

Figure 10 shows the same voxels, coloured by the value of W_{static} , which also takes into account the ray tracing results. The tree from Fig. 9 was cut out to show the complete facade. Here, the blue colour represents a small value for W_{static} and red a high value. Both, the roof and the facade parts, which are partly covered by trees (as shown in Fig. 9), have a high value. In contrast, the scaffold, which only appeared during approximately 3–5 measurements, has a much smaller value because the corresponding voxels were traversed by the laser beam during the other measurement runs and could thus be identified as being empty.

By selecting a threshold value of 0.8, outliers and temporary objects can be removed quite reliably. What remain are the voxels which have a very high probability of belonging to static objects, as shown in Fig. 11. This threshold can be modified according to the class or other external conditions.

For the classes “vegetation” and “traffic sign”, the occupancy states from the ray tracing were not used, because they are not available for many voxels of these classes. The reason for this is the pre-processing of the scan strip data, where small segments are discarded, as described in Sect. 4.3. In the future, this could be fixed by including small segments as well.

Figure 12 shows the created reference map for the measurement area “Nordstadt”. The extracted ground segments

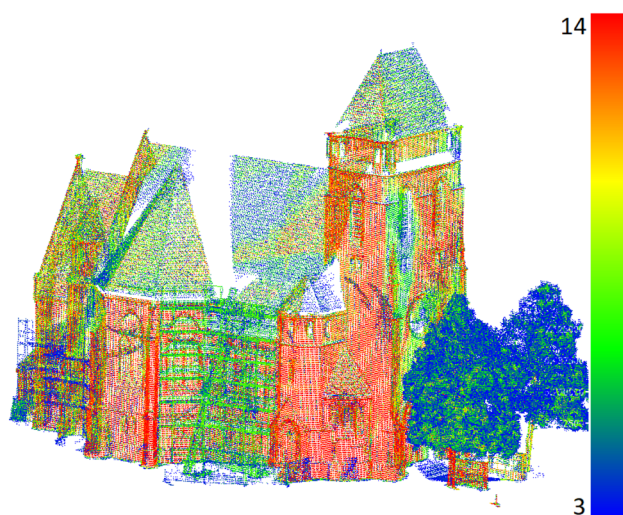


Fig. 9 Example voxel grid of the “Lutherkirche”, coloured by the number of occurrences during 14 epochs of the measurement campaign

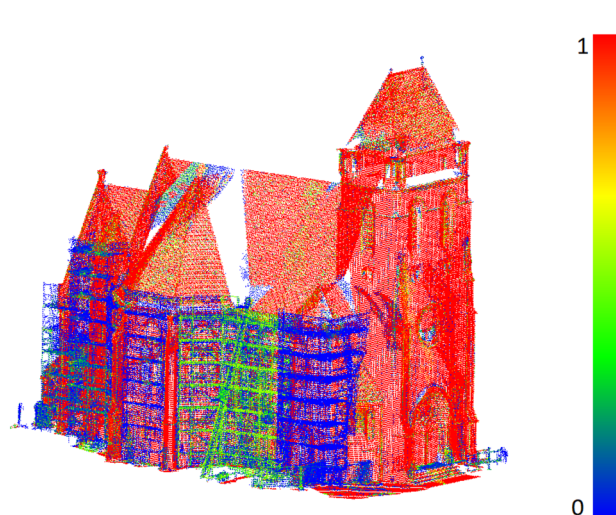


Fig. 10 Example voxel grid of the “Lutherkirche”, coloured by W_{static} from 0 to 1 (the tree from Fig. 9 was cut out to show the complete facade)

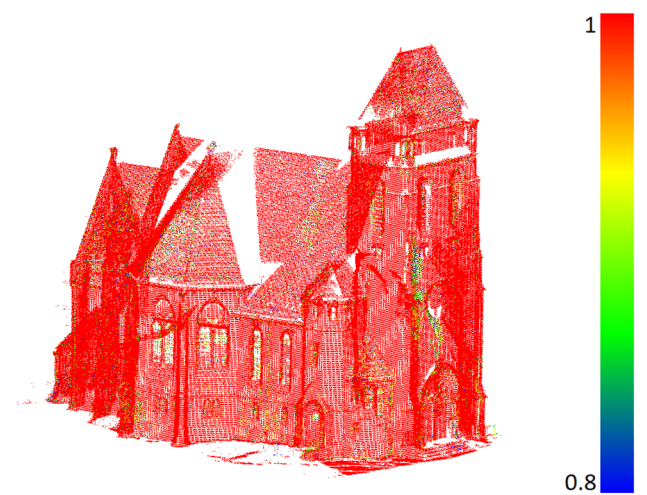


Fig. 11 Example voxel grid of the “Lutherkirche”, coloured by W_{static} from 0.8 to 1 (the tree from Fig. 9 was cut out to show the complete facade)

are coloured in purple. Some visible holes on the street are caused by parked cars, which occluded the ground during the corresponding measurement. The classes “vegetation” (dark green), “traffic sign” (yellow) and “pole” (orange) are only included in the map if their number of occurrences is at least three. The classes “facade” (grey) and “fence”(brown) are included if their value for W_{static} is at least 0.8.

6 Conclusion and Outlook

6.1 Summary of the Results

The aim of this work was to create a static reference map of an urban environment. Therefore, LiDAR point clouds were segmented and classified. Several measuring epochs were compared to separate between static and dynamic objects as well as to detect changes. The result is a 3D voxel grid that stores all static objects and their classification result. In the following, the individual steps, experienced difficulties, and possible solutions are summarised.

For the segmentation, we used a region growing algorithm, which separates objects by a distance threshold. For most cases this works well. However, if various ground levels occur in a point cloud, our implementation only considers the lowest one. This could be easily avoided by a more elaborated ground segmentation, e.g. by searching for several horizontal planes within a certain height range, as suggested by Paul et al. (2012). We also do not consider any slope so far, which could be improved by estimating the topography of the scene, as in Blomley and Weinmann (2017).

The classification yields very good results with an overall accuracy of 94%. Due to the low number of training examples and strong similarity of some classes, the original

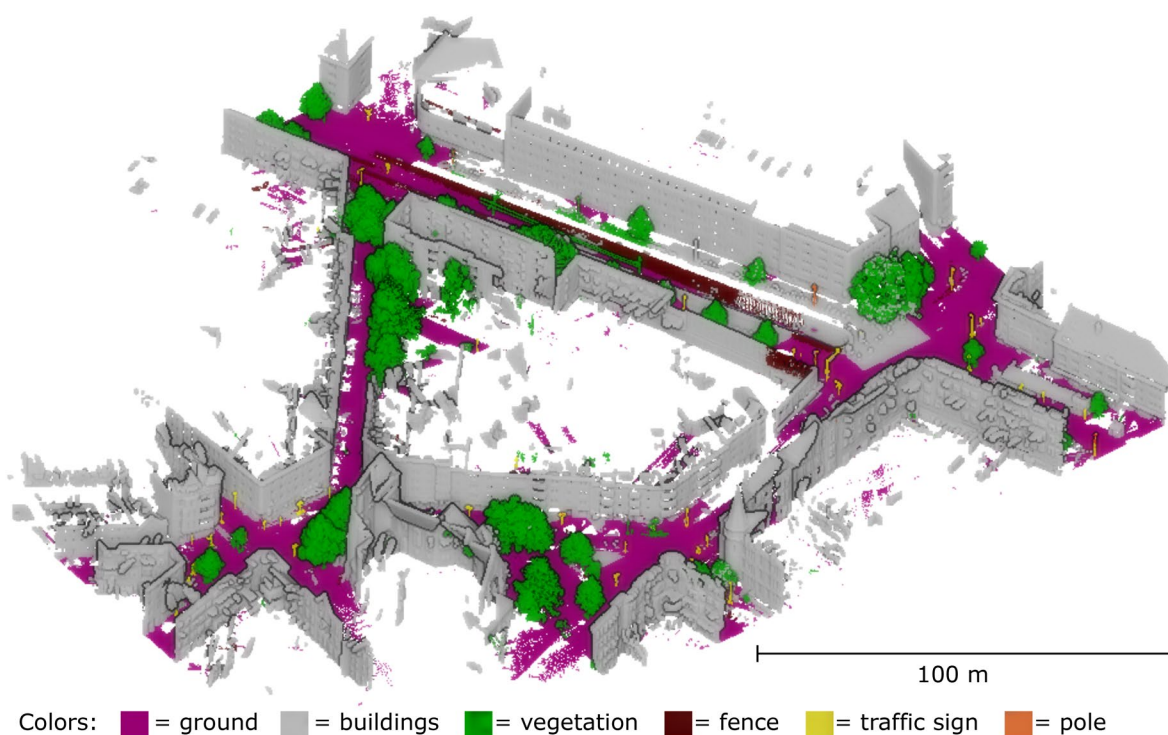


Fig. 12 Reference map of the Nordstadt measurement area

classes from the Cityscapes dataset (Cordts et al. 2016) had to be reduced significantly. Our random forest-based classification shows good results, also in comparison to similar work, e.g. Lehtomäki et al. (2016) and Paul et al. (2012). The remaining misclassifications are mainly caused by objects inside buildings, measured through the windows. These objects are often incomplete or do not belong to any of the used classes. One way to deal with this would be to detect the facades first and delete all objects behind them, before the main classification is invoked. Another factor that leads to misclassifications are merged objects which were not separated by the region growing algorithm, e.g. traffic signs ingrown by trees, bicycles attached to traffic signs or leaned against facades, etc.

A major advantage of our approach is that we combine classification and observation sequences. This is especially important when we extract the static part of the environment. “Vegetation”, for example, is defined as static but changes over time due to growth, seasonal changes or human intervention. Contrary to this is the class “vehicle” which is defined as dynamic but could be assumed to be static in areas of permanently parked cars. If only the observation sequences were used for filtering the static objects, it can happen that both classes are treated similarly, although this is unintended. Classification thus enables a deeper understanding of the behaviour of objects, which is self-evident for humans.

A visual inspection of the reference map shows that it provides very good results. Objects such as awnings, scaffolds or other segments associated with static objects that only occur during a few measurement runs are not added to the static map. If a previously static object disappears, e.g. due to a building demolition, it is also removed from the static map after a certain number of observations.

We envision many possible applications for our reference map. Of course, distinctive points in the map can support localisation, as e.g. mentioned in Schlichting and Brenner (2016b). However, the reference map is useful in a much more broader sense, as the combination of semantic and temporal information allows an automated system (using this map) to anticipate the state of the environment at a given point in time more precisely.

6.2 Outlook

In future work, we want to use larger amounts of data on the one hand and improve data processing on the other hand. For the former, we need to create more training data, especially for the smaller classes, in order to improve the classification results and to introduce more classes. In addition, the methods developed in this work will be improved and then applied to all data of the measurement campaign. In the data

processing part, the ground removal and facade detection could be improved.

Another important next step is the incremental integration of new measurement data. For this, the definition of W_{static} has to be extended in such a way that newly recorded data has a higher influence than older data, so that changes, such as new or demolished buildings, appear faster in the reference map. In this context it would also be interesting to gain insight into the update rate which is required to maintain a useful reference map, because the different real-world object classes each show their own temporal behaviour.

Acknowledgements Julia Schachtschneider was supported by the German Research Foundation (DFG), as part of the Research Training Group i.c.sens, GRK 2159, ‘Integrity and Collaboration in Dynamic Sensor Networks’. The long term measurement campaign used in this paper was also conducted within the scope of this project.

Funding Open Access funding enabled and organized by Projekt DEAL.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Aijazi AK, Checchin P, Trassoudaine L (2013) Detecting and updating changes in lidar point clouds for automatic 3d urban cartography. *ISPRS Ann Photogramm Remote Sens Spat Inf Sci I-5/W2:7–12*. <https://doi.org/10.5194/isprsannals-II-5-W2-7-2013>
- Blanco-Claraco JL, Moreno-Dueñas FÁ, González-Jiménez J (2013) The Málaga urban dataset: high-rate stereo and lidar in a realistic urban scenario. *Int J Robot Res* 33(2):207–214. <https://doi.org/10.1177/0278364913507326>
- Blomley R, Weinmann M (2017) Using multi-scale features for the 3d semantic labelling of airborne laser scanning data. *ISPRS Ann Photogramm Remote Sens Spat Inf Sci IV-2/W4:43–50*. <https://doi.org/10.5194/isprs-annals-IV-2-W4-43-2017>
- Breiman L (1984) Classification and regression trees. The Wadsworth statistics/probability series, Wadsworth International Group and Wadsworth & Brooks/Cole and Wadsworth & Brooks/Cole Advanced Books & Software, Belmont, Calif. and Pacific Grove, Calif. and Pacific Grove, Calif. and Monterey, Calif
- Breiman L (1996) Technical note: some properties of splitting criteria. *Mach Learn* 24:41–47
- Brenner C (2016) Scalable estimation of precision maps in a MapReduce framework. In: Proceedings of the 24th ACM SIGSPATIAL international conference on advances in geographic information systems, SIGSPATIAL ’16. ACM, New York, pp 27:1–27:10. <https://doi.org/10.1145/2996913.2996990>

- Carlevaris-Bianco N, Ushani AK, Eustice RM (2016) University of Michigan north campus long-term vision and lidar dataset. *Int J Robot Res* 35(9):1023–1035. <https://doi.org/10.1177/0278364915614638>
- Cordts M, Omran M, Ramos S, Rehfeld T, Enzweiler M, Benenson R, Franke U, Roth S, Schiele B (2016) The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pp 3213–3223
- Demantke J, Mallet C, David N, Vallet B (2011) Dimensionality based scale selection in 3d lidar point clouds. *Int Arch Photogramm Remote Sens Spat Inf Sci* 38(5/W12):97–102
- Eckstein L, Form T, Maurer M, Schöneburg R, Spiegelberg G, Stiller C (2018) Automatisiertes Fahren: VDI-Status report Juli 2018
- Engelmann F, Kontogianni T, Hermans A, Leibe B (2017) Exploring spatial context for 3d semantic segmentation of point clouds. In: Proceedings of the IEEE international conference on computer vision workshops, pp 716–724
- Fuse T, Yokozawa N (2017) Development of a change detection method with low-performance point cloud data for updating three-dimensional road maps. *ISPRS Int J Geo-Inf* 6(12):398. <https://doi.org/10.3390/ijgi6120398>
- Gehring J, Hebel M, Arens M, Stilla U (2017) An approach to extract moving objects from mls data using a volumetric background representation. *ISPRS Ann Photogramm Remote Sens Spat Inf Sci IV-1/W1:107–114*. <https://doi.org/10.5194/isprs-annals-IV-1-W1-107-2017>
- Geiger A, Lenz P, Stiller C, Urtasun R (2013) Vision meets robotics: the kitti dataset. *Int J Robot Res* 32(11):1231–1237. <https://doi.org/10.1177/0278364913491297>
- Grilli E, Menna F, Remondino F (2017) A review of point clouds segmentation and classification algorithms. *ISPRS Int Arch Photogramm Remote Sens Spat Inf Sci XLII-2/W3:339–344*. <https://doi.org/10.5194/isprsarchives-XLII-2-W3-339-2017>
- Hackel T, Savinov N, Ladicky L, Wegner JD, Schindler K, Pollefeys M (2017) Semantic3d.net: a new large-scale point cloud classification benchmark. *ISPRS Ann Photogramm Remote Sens Spat Inf Sci IV-1/W1:91–98*. <https://doi.org/10.5194/isprs-annals-IV-1-W1-91-2017>
- Hebel M, Arens M, Stilla U (2013) Change detection in urban areas by object-based analysis and on-the-fly comparison of multi-view als data. *ISPRS J Photogramm Remote Sens* 86:52–64. <https://doi.org/10.1016/j.isprsjprs.2013.09.005>
- Himmelsbach M, Luettel T, Wuensche H (2009) Real-time object classification in 3d point clouds using point feature histograms. In: 2009 IEEE/RSJ international conference on intelligent robots and systems, pp 994–1000
- Johnson AE, Hebert M (1999) Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Trans Pattern Anal Mach Intell* 21(5):433–449
- Landrieu L, Raguette H, Vallet B, Mallet C, Weinmann M (2017) A structured regularization framework for spatially smoothing semantic labelings of 3d point clouds. *ISPRS J Photogramm Remote Sens* 132:102–118
- Landrieu L, Simonovsky M (2018) Large-scale point cloud semantic segmentation with superpoint graphs. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4558–4567
- Lehtomäki M, Jaakkola A, Hyypä J, Lampinen J, Kaartinen H, Kukko A, Puttonen E, Hyypä H (2016) Object classification and recognition from mobile laser scanning point clouds in a road environment. *IEEE Trans Geosci Remote Sens* 54(2):1226–1239. <https://doi.org/10.1109/TGRS.2015.2476502>
- Lindenbergh R, Pietrzyk P (2015) Change detection and deformation analysis using static and mobile laser scanning. *Appl Geomat* 7(2):65–74. <https://doi.org/10.1007/s12518-014-0151-y>
- Maddern W, Pascoe G, Linegar C, Newman P (2016) 1 year, 1000 km: the oxford robotcar dataset. *Int J Robot Res* 36(1):3–15. <https://doi.org/10.1177/0278364916679498>
- OpenStreetMap Foundation (2019) Openstreetmap. <https://www.openstreetmap.org/>
- Pandey G, McBride JR, Eustice RM (2011) Ford campus vision and lidar data set. *Int J Robot Res* 30(13):1543–1552. <https://doi.org/10.1177/0278364911400640>
- Paul R, Triebel R, Rus D, Newman P (2012) Semantic categorization of outdoor scenes with uncertainty estimates using multi-class Gaussian process classification. In: Proceedings of the IEEE international conference on robotics and automation (ICRA), pp 2404–2410. <https://doi.org/10.1109/IROS.2012.6386073>
- Qi CR, Su H, Mo K, Guibas LJ (2017) Pointnet: deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 652–660
- Qin R, Tian J, Reinartz P (2016) 3d change detection—approaches and applications. *ISPRS J Photogramm Remote Sens* 122:41–56. <https://doi.org/10.1016/j.isprsjprs.2016.09.013>
- RIEGL Laser Measurement Systems (2012) Compact mobile laser scanning system riegli vmx-250: data sheet
- Roynard X, Deschaud JE, Goulette F (2018) Paris-lille-3d: a large and high-quality ground-truth urban point cloud dataset for automatic segmentation and classification. *Int J Robot Res* 37(6):545–557. <https://doi.org/10.1177/0278364918767506>
- Rusu RB, Bradski G, Thibaux R, Hsu J (2010) Fast 3d recognition and pose using the viewpoint feature histogram. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems. IEEE, Piscataway, pp 2155–2162. <https://doi.org/10.1109/IROS.2010.5651280>
- Schachtschneider J, Brenner C (2020) Creating multi-temporal maps of urban environments for improved localization of autonomous vehicles. *ISPRS Int Arch Photogramm Remote Sens Spat Inf Sci XLIII-B2-2020:317–323*. <https://doi.org/10.5194/isprs-archives-XLIII-B2-2020-317-2020>
- Schachtschneider J, Schlichting A, Brenner C (2017) Assessing temporal behavior in lidar point clouds of urban environments. *ISPRS Int Arch Photogramm Remote Sens Spat Inf Sci XLII-1/W1:543–550*. <https://doi.org/10.5194/isprs-archives-XLII-1-W1-543-2017>
- Schlichting A, Brenner C (2016) Generating a hazard map of dynamic objects using lidar mobile mapping. *Photogramm Eng Remote Sens* 82(12):967–972. <https://doi.org/10.14358/PERS.82.12.967>
- Schlichting A, Brenner C (2016b) Vehicle localization by lidar point correlation improved by change detection. *ISPRS Int Arch Photogramm Remote Sens Spat Inf Sci XLI-B1:703–710*. <https://doi.org/10.5194/isprsarchives-XLI-B1-703-2016>
- Serna A, Marcotegu B, Goulette F, Deschaud JE (2014) Paris-rue-madame database: a 3d mobile laser scanner dataset for benchmarking urban detection, segmentation and classification methods. *Appl Methods ICPRAM* 2014:819–824
- Sokolova M, Lapalme G (2009) A systematic analysis of performance measures for classification tasks. *Inf Process Manag* 45(4):427–437. <https://doi.org/10.1016/j.ipm.2009.03.002>
- Vallet B, Brédif M, Serna A, Marcotegui B, Papanoditis N (2015) Terrestrial urban point cloud analysis benchmark. *Comput Graph* 49:126–133. <https://doi.org/10.1016/j.cag.2015.03.004>
- Weinmann M, Jutzi B, Hinz S, Mallet C (2015) Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers. *ISPRS J Photogramm Remote Sens* 105:286–304
- Xiao W, Vallet B, Brédif M, Papanoditis N (2015) Street environment change detection from mobile laser scanning point clouds. *ISPRS J Photogramm Remote Sens* 107:38–49. <https://doi.org/10.1016/j.isprsjprs.2015.04.011>