PFG (2017) 85:41–52
DOI 10.1007/s41064-016-0003-y

ORIGINAL ARTICLE

Efficient Online Segmentation for Sparse 3D Laser Scans

Igor Bogoslavskyi[1] · **Cyrill Stachniss**[1]

Received: 1 July 2016 / Accepted: 1 December 2016 / Published online: 20 February 2017

Abstract The ability to extract individual objects in the scene is key for a large number of autonomous navigation systems such as mobile robots or autonomous cars. Such systems navigating in dynamic environments need to be aware of objects that may change or move. In most perception cues, a pre-segmentation of the current image or laser scan into individual objects is the first processing step before a further analysis is performed. In this paper, we present an effective method that first removes the ground from the scan and then segments the 3D data in a range image representation into different objects. A key focus of our work is a fast execution with several hundred Hertz. Our implementation has small computational demands so that it can run online on most mobile systems. We explicitly avoid the computation of the 3D point cloud and operate directly on a 2.5D range image, which enables a fast segmentation for each 3D scan. This approach can furthermore handle sparse 3D data well, which is important for scanners such as the new Velodyne VLP-16 scanner. We implemented our approach in C++ and ROS, thoroughly tested it using different 3D scanners, and will release the source code of our implementation. Our method can operate at frame rates that are substantially higher than those of the sensors while using only a single core of a mobile CPU and producing high-quality segmentation results.

Keywords Segmentation · 3D laser · Online · Range image · Sparse data · Point cloud

✉ Igor Bogoslavskyi
igor.bogoslavskyi@uni-bonn.de

Cyrill Stachniss
cyrill.stachniss@igg.uni-bonn.de

[1] Institute of Geodesy and Geoinformation, University of Bonn, Nussallee 15, 53115 Bonn, Germany

Zusammenfassung *Effiziente Online-Segmentierung für schwach besetzte 3D-Laserscans*. Die schnelle und vollautomatische Interpretation eine Szene spielt beim Einsatz autonomer Autos oder mobiler Roboter eine zentrale Rolle und wird in nahezu allen dynamischen Umgebungen benötigt. Der erste Schritt eines typischen Perzeptionssystemes zur Szeneninterpretation ist häufig die Segmentierung der Szene in einzelne Bestandteile. In dieser Arbeit stellen wir ein effizientes Segmentierungsverfahren für 3D Laserscanner vor, welches mit mehreren 100 Hz auf handelsüblichen CPUs ausgeführt werden kann und gleichzeitig hochwertige Ergebnisse liefert. Wir erreichen die schnelle Verarbeitung, indem Berechnungen auf 3D Punktwolken vermieden und statt dessen direkt auf 2.5D-Entfernungsbildern durchgeführt werden. Neben der schnellen Berechnung kann so auch mit niedrig aufgelösten Laserscans gut umgegangen werden. Wir haben unseren Ansatz in C++ und ROS implementiert und mit verschiedenen Datensätzen evaluiert. Es zeigt sich, dass unser Verfahren die Laserdaten deutlich schneller verarbeitet als typische Laserscanner diese erzeugen und gleichzeitig eine qualitativ hochwertige Segmentierung der Szene liefert.

1 Introduction

Image segmentation in RGB and multi-spectral data is a common problem in photogrammetric image analysis, computer vision, and remote sensing. Separating individual objects in 3D laser range data is also an important task for autonomous navigation of mobile robots or instrumented cars. An autonomous vehicle that is navigating in an unknown environment faces the complicated task of reasoning about its surroundings, see Golovinskiy and Funkhouser (2009), Hebel and Stilla (2008), Himmelsbach et al. (2010), Küm-

Fig. 1 *Left* Segmentation of objects such as people, cars, and trees generated from sparse 3D range data recorded with Velodyne VLP-16 scanner. *Colours* correspond to different segments. *Right* Clearpath Husky robot used for the experiments

merle et al. (2013), Steinhauser et al. (2008), Teichman and Thrun (2012), Wang and Shan (2009), Wurm et al. (2008). There might be objects that constrain the possible actions of the robot or that may interfere with the robot's own plans. Thus, the interpretation of the robot's surroundings is key for robust operation. While some approaches focus on finding specific objects in a dynamic scene (Hanel et al. 2015; Menze et al. 2015; Leibe et al. 2008), most perception pipelines perform a segmentation of the environment into individual objects before a further interpretation is performed. Therefore, we see the need for an efficient online segmentation approach for 3D range data as this allows the robot to directly react to individual objects in its surroundings. This segmentation should be available in real time as the system needs to reason about what it sees right when the data become available to react appropriately.

Object segmentation from raw sensor data is especially relevant when mapping or operating in dynamic environments. In busy streets with cars and pedestrians, for example, the maps can be influenced by wrong data associations caused by the dynamic nature of the environment. A key step to enable a better reasoning about such objects and to potentially neglect dynamic objects during scan registration and mapping is the segmentation of the 3D range data into different objects so that they can be tracked separately, see Dewan et al. (2016).

Besides rather expensive terrestrial laser scanners, there are also less accurate and cheaper scanners targeted at mobile robotics applications. One example is the 16-beam LIDAR by Velodyne, which is becoming increasingly more popular and can be installed on relatively low-cost platforms. If we compare the data provided by the 16-beam LIDAR with those provided by the 64-beam variant or even a terrestrial scanner, we observe a substantial drop in the vertical angular resolution. This poses several challenges to a segmentation algorithm operating on such 3D data. Sparser point clouds lead to an increased Euclidean distance between neighbouring points even if they stem from the same object. Thus, these sparse 3D points render it more difficult to reason about segments. The situation becomes even harder with the increase in distance between the object and the sensor.

The contribution of this paper is a robust method for separating ground from the rest of the scene and a fast and effective segmentation approach for 3D range data obtained from modern laser range finders such as Velodyne scanners. To achieve the final segmentation, we first perform a robust ground separation which can detect ground fast and reliably. In contrast to several other approaches, the ground can have slight curvature and does not necessarily have to be entirely flat. We also do not use any kind of sub-sampling and decide for each pixel of the range image whether it belongs to ground or not. An example of our segmentation with ground removed is depicted in Fig. 1 where people and cars are correctly segmented using data from a Velodyne VLP-16 scanner.

Our segmentation method provides meaningful segmentations and runs multiple times faster than the acquisition of the scan. Even on a mobile CPU, we can process the scans of a Velodyne with over 70 Hz (64 beams) or 250 Hz (16 beams) and thus faster than the scans are acquired. We achieve this by performing all computations on a cylindrical range image. This method is advantageous, as the range image is often small, dense, and maintains the neighbourhood information implicitly. Moreover, our approach is suited for scanners that provide comparably sparse point clouds as these clouds can still be represented as a dense range image.

This paper extends our recently published conference paper on 3D range data segmentation (Bogoslavskyi and Stachniss 2016). In this work, we added the robust ground removal and provide an extended experimental evaluation.

2 Related Work

Segmenting objects from 3D point clouds is a relatively well-researched topic. There is substantial amount of work that targets acquiring a global point cloud and segmenting it off-line. Examples for such approaches are the works by Abdullah et al. (2014), Endres et al. (2009), Golovinskiy and Funkhouser (2009), Hebel and Stilla (2008) and Wang and Shan (2009). These segmentation methods have been used on a variety of different data produced by 3D range sensors or 2D lasers in push-broom mode. The photogram-

metric society has also been active in the field of segmenting big point clouds into different objects. Velizhev et al. (2012) focus on learning the classes of the objects and detecting them in huge point clouds via a voting-based method. These point clouds can be large, and the work by Hackel et al. (2016) targets the runtime along with the quality of classification. In contrast with these works, we focus on the segmentation of range data that come from a 3D laser scanner such as a Velodyne that provides a 360 degree field of view in a single scan and is used for online operation on a mobile robot. Additionally, we target segmentation of a scene without the knowledge about the objects in it and without any prior learning and not using complex features. For a comprehensive analysis of methods that perform supervised scene segmentation we refer the reader to Weinmann et al. (2015).

Ground removal is an often used pre-processing step and is, therefore, well discussed in the literature. There are a number of papers that use RANSAC for fitting a plane to the ground and removing points that are near this plane such as the work by Ošep et al. (2016). Another prominent method of ground detection is a side-product of full semantic segmentation of the scene, where all parts of the scene get a semantic label. The ground is then segmented as one class; for more details we refer the reader to the papers by Hermans et al. (2014) and Bansal et al. (2009). A couple of approaches use a 2D-grid and analyse the heights of the points that fall into its bins, taking decisions about points being parts of the ground based on this information. The decisions can be taken based on the inclination of lines between consecutive cells as in works by Petrovskaya and Thrun (2008) and Leonard et al. (2008) or by analysing the height above the lowest local point as in works by Gorte et al. (2015) and Behley et al. (2013).

Segmentation techniques for single scans without requiring additional information can be divided into three groups. The first group, represented by the works by Douillard et al. (2011, 2014), performs the segmentation in the 3D domain by defining sophisticated features that explain the data in 3D or by removing the ground plane and segmenting the clouds with a variant of a nearest neighbour approach as shown by Choe et al. (2012) and Klasing et al. (2008). Feature-based approaches, while allowing for accurate segmentation, are often comparably time consuming and may limit the application for online applications to a robot with substantial computational resources.

The second group focuses on projecting 3D points onto a 2D grid positioned on the ground plane. The segmentation is then carried out on occupied grid cells as in studies by Behley et al. (2013), Himmelsbach et al. (2010), Korchev et al. (2013) and Steinhauser et al. (2008). These algorithms are fast and suitable to run online. Quite often, however, they have a slight tendency to under-segment the point clouds, i.e. multiple objects may be grouped as being one object if they are close to each other. This effect often depends on the choice of the grid discretisation, so that the grid width may need to be tuned for individual environments. Additionally, some of these approaches can suffer from under-segmenting objects in the vertical direction.

The third group of approaches performs the segmentation on a range image and our approach belongs to this group of techniques. For example, Moosmann et al. (2009) and Moosmann (2013) use a range image to compute local convexities of the points in the point cloud. In contrast to that, our approach avoids computing complex features and, thus, is easier to implement, runs very fast and produces comparable results. We, therefore, believe that our approach is a valuable contribution to a vast and vibrant field of 3D point cloud segmentation, and consequently we will contribute our approach to the open source ROS community by providing the source code for our implementation.

There are also several works (Pylvanainen et al. 2010; Strom et al. 2010) that perform segmentation on RGBD data acquired from a LIDAR registered with a camera. Registering one or multiple cameras with the laser scanner requires a more sophisticated setup and the segmentation becomes more demanding. Using both cues may improve the results but it is seldom possible at speeds faster than the frame rate. Therefore, we focus on segmenting unknown objects from pure 3D range data not requiring any additional visual or intensity information.

Visual information is not the only information that aids segmentation. Temporal information and tracking are also shown to be useful to enhance the segmentation performance by Floros and Leibe (2012) and Teichman and Thrun (2012). While the benefit of using the information about the moving objects is clear, we show that it is possible to perform a fast and meaningful segmentation on single scans even without relying on temporal integration.

3 Range Image-Based Ground Removal

Before performing object segmentation, we remove the ground from the scan. A standard approach to ground removal simply discards all 3D points that are lower than the vehicle (assuming we know where the sensor has been mounted on the mobile base/robot). While this approach may work in simple scenes, it fails if the vehicle's pitch or roll angle is unequal to zero or if the ground is not a perfect plane. Using RANSAC-based plane fitting may improve the situation but even using this method, non-zero curvatures may remain a challenge and the operation can be time consuming. Thus, we take a different approach.

Most laser range scanners provide raw data in the form of individual range readings per laser beam with a time stamp and an orientation of the beam. This allows us to directly convert the data into a range image. The number of rows in the

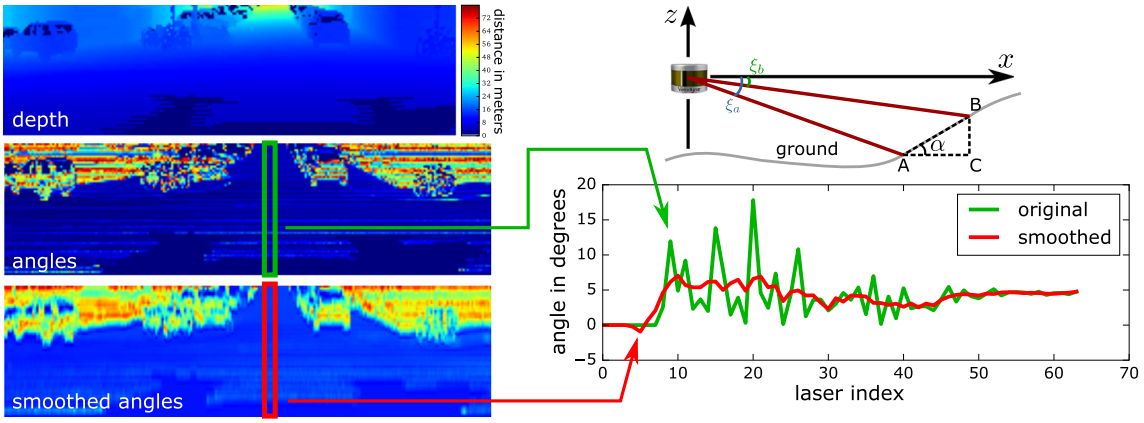

Fig. 2 *Top left* Part of a range image. *Middle left* An image generated by showing α angles. *Bottom left* Angles after applying the Savitsky–Golay smoothing. *Top right* An illustration of α angle. *Bottom right* Illustration of the smoothing for a column of α angles as marked in the *left* image

image is defined by the number of beams in the vertical direction, i.e. 16, 32 or 64 for the Velodyne scanners. The number of columns is given by the range readings per 360° revolution of the laser. Each pixel of such a virtual image stores the measured distance from the sensor to the object. To speed up computations, one may even consider to combine multiple readings in the horizontal direction into one pixel if needed.

In our implementation, we use the-above described range images and construct them directly from the raw measurements of the laser scanner, not computing the 3D point cloud. In case, however, a different laser scanner or a different device driver is used that only provides a 3D point cloud per revolution and not the individual range measurements, one can project the 3D points cloud onto a cylindrical image, compute the Euclidean distance per pixel, and proceed with our approach. This will increase the computational demands by approximately a factor of 2 for the whole approach but still allows for a comparably fast segmentation.

For identifying the ground plane, we make three assumptions. First, we assume that the sensor is mounted roughly horizontally on the mobile base/robot (this assumption can be relaxed, but the explanation would turn out to be more complex). Second, we assume that the curvature of the ground is low. Third, we assume that the robot observes the ground plane at least in some pixels of the lowest row of the range image (corresponding to the laser beam scans close to the ground close to the robot).

With these assumptions in place we start by turning each column c of the range image R into a stack of angles $\alpha^r_{r-1,c}$, where each of these angles represents the angle of inclination of a line connecting two points A and B derived from two range readings $R_{r-1,c}$ and $R_{r,c}$ in neighbouring rows $r-1, r$ of the range image, respectively, as depicted in the top right part of Fig. 2. Knowing two range readings of vertically consecutive individual laser beams, we can compute the angle α using trigonometric rules as follows:

$$\begin{aligned} \alpha &= \operatorname{atan2}(\|BC\|, \|AC\|) = \operatorname{atan2}(\Delta z, \Delta x) \\ \Delta z &= |R_{r-1,c} \sin \xi_a - R_{r,c} \sin \xi_b| \\ \Delta x &= |R_{r-1,c} \cos \xi_a - R_{r,c} \cos \xi_b| \end{aligned} \tag{1}$$

where ξ_a and ξ_b are vertical angles of the laser beams corresponding to rows $r-1$ and r.

Note that we need two range readings for each α computation and so the size of the stack of α angles has size one less than the number of rows in the range image. We then treat all stacks of these angles as a matrix $M_\alpha = [\alpha^r_{r-1,c}]$, where r and c are row and column coordinates of the corresponding range readings from the range image.

Unfortunately, LIDAR sensors such as the Velodyne HDL-64 produce a substantial amount of outliers in the range measurements, discussed in more detail in the work of Leonard et al. (2008), which impacts the computation of the angle α in Fig. 2. We, therefore, need a way to eliminate such outliers. Weinmann and Jutzi (2015) address this problem by computing features over a small local neighbourhood of every pixel of a range image to detect if a reading can be treated as reliable or not. This approach filters out unreliable readings but also the points on the borders of the objects. As these points are important for performing segmentation, we instead compute the corresponding angles from all available data points and smooth the computed angles afterwards. To achieve such smoothing, we apply the Savitsky–Golay filter to every column of M_α. This filter performs least-squares optimization to fit a local polynomial for a given window size to the data. In their work, Savitzky and Golay (1964) show that one can avoid the explicit least-squared fitting of the polynomials and compute an effective approximation relying on precomputed coefficients, which allows for greater computational efficiency.

We carry out the ground labelling on the matrix M_α after applying the Savitsky–Golay filter to its columns starting

with the entries that we expect to belong to the ground and labelling similar components together using breadth-first search. Breadth-first search (BFS) is a popular graph search or traversal algorithm. It starts at a given node of the graph and explores the directly neighbouring nodes first, before moving to the next level of neighbours. In our approach, we consider the difference in the calculated angles α over an N4 neighbourhood on a grid to decide if two neighbouring elements of the matrix M_α should be labelled together by the breadth-first search. For that purpose, we select a threshold Δa, set to 5° in our experiments.

We start by labelling each element of the lowest row as ground if the corresponding $\alpha^1_{0,c}$ is smaller than a predefined angle (45° in our current implementation), i.e. we are not labelling any almost vertical objects such as walls. Let the set G be a set of all column indices in the first row that we have labelled as ground.

Algorithm 1 Ground Labelling

```
1:  procedure LabelGround(R)
2:      M ← [α^r_{r−1,c}], matrix of angles α computed with Eq. (1).
3:      for c = 1 ... R_cols do
4:          if M(0, c) not labelled then
5:              LabelGroundBFS(0, c);
6:  procedure LabelGroundBFS(r, c)
7:      queue.push({r, c})
8:      while queue is not empty do
9:          {r, c} ← queue.top()
10:         {r, c} ←labelled as ground
11:         for {r_n, c_n} ∈ neighbourhood{r, c} do
12:             if |M(r, c) − M(r_n, c_n)| < 5° then
13:                 queue.push({r_n, c_n})
14:         queue.pop()
```

For every $c \in G$ we label the connected component using BFS starting from $\alpha^1_{0,c}$ as ground as depicted in procedure `LabelGround` in Algorithm 1. By the time we have processed all $c \in G$, all the ground pixels in the image have been labelled as such. Figure 3 shows an example point cloud with the ground detected by our algorithm marked in light blue.

4 Fast and Effective Segmentation Using Laser Range Images

This work focuses on fast 3D range data segmentation for online processing on a mobile robot that is equipped with a rotating scanner such as one of the three popular Velodyne scanners with 16, 32, or 64 beams. The vertical resolution of the sensors has an impact on the difficulty of the segmentation problem. For every pair of neighbouring points, one basically has to decide if the laser beams have been reflected by the same object or not.

In our approach, outlined in Fig. 4, we avoid the explicit creation of the 3D point cloud and perform our computations using a laser range image, in our case a cylindrical one for the Velodyne scanners. This has two advantages: first, we can exploit the clearly defined neighbourhood relations directly in the range image and this makes the segmentation problem easier. Second, we avoid the generation of the 3D point cloud, which makes the overall approach faster to compute.

We assume the vehicle to move on the ground (see Fig. 1 for our setup) and we expect the sensor to be oriented roughly horizontally with respect to the wheels. Thus, we can quickly obtain an estimate of the ground plane by analysing the columns of such range image as described in Sect. 3. The ground is then removed from the range image.

The key component of our approach is the ability to estimate which measured points originate from the same object for any two laser beams. We explicitly avoid feature computation and work with raw sensor data, taking a decision for each point of the 3D range data.

We present an easy to implement and fast to compute but yet effective approach to find the components that belong to one object. To answer the question if two laser measurements belong to the same object, we use an angle-based measure, which is illustrated in Fig. 5 and is described in the following paragraphs.

The left image of Fig. 5 shows an example scene with two people walking close to each other in front of a cyclist, who passes between them and a parked car. This scene has been

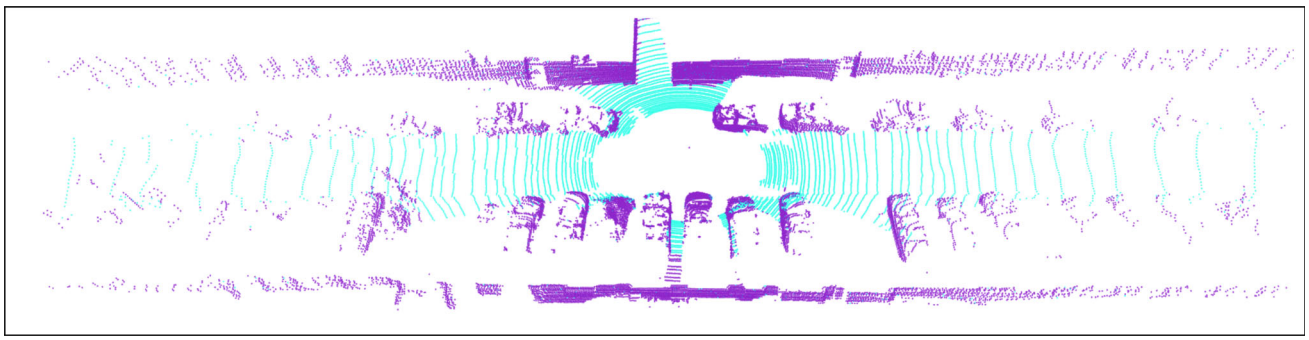

Fig. 3 An example scene seen from above with ground marked *light blue*

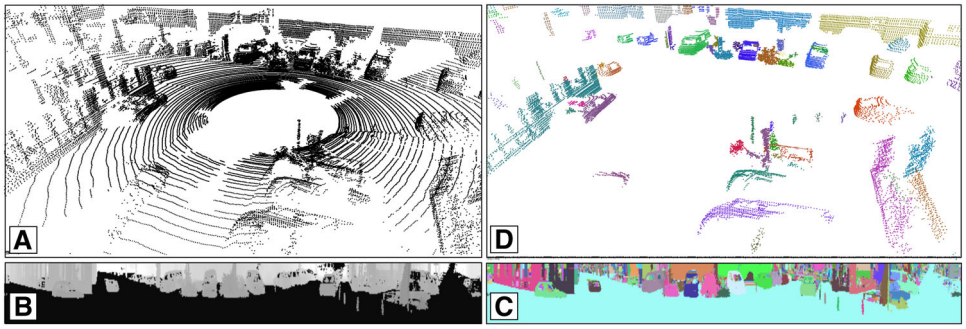

Fig. 4 Illustration of our method. **a** Point cloud from Velodyne, which is shown for illustration reasons only. **b** We build up a range image not considering points lying on the ground plane and **c** perform the segmentation in the range image directly. **d** This allows us to provide individual small point clouds for the different segments. The different objects are shown with random *colours*. Range and label images are scaled for better visibility

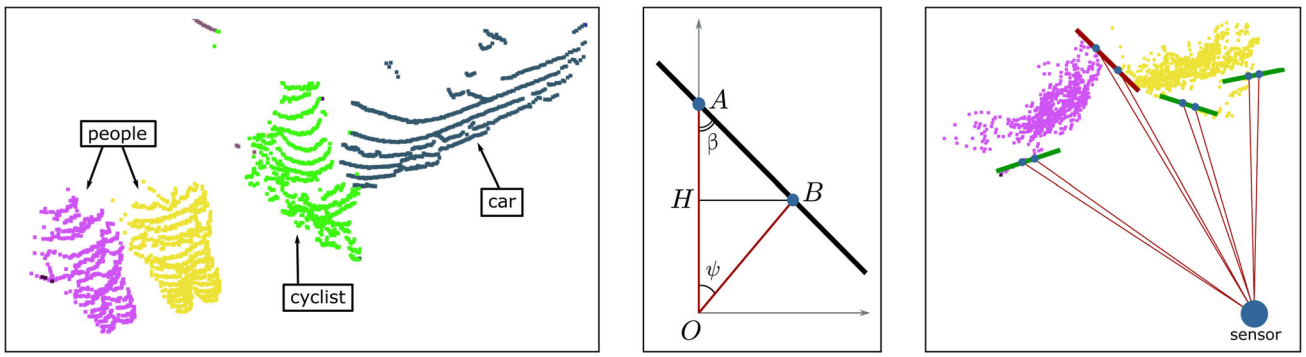

Fig. 5 *Left* Example scene with two pedestrians, a cyclist and a car. *Middle* Given that the sensor is in O and the lines OA and OB represent two laser beams, the points A and B spawn a *line* that estimates the surface of an object should they both belong to the same object. We make the decision about this fact based on the angle β. If $\beta > \theta$, where θ is a predefined threshold, we consider the points to represent one object. *Right* A *top* view on the pedestrians from the example scene. The *green lines* represent points with $\beta > \theta$ while the *red* one shows an angle that falls under the threshold and thus labels objects as different

recorded using our Velodyne VLP-16 scanner. The middle image shows an illustration of two arbitrary points A and B measured from the scanner located at O with the illustrated laser beams OA and OB. Without loss of generality, we assume the coordinates of A and B to be in a coordinate system which is centred in O and the y-axis is oriented along the longer of two laser beams. We define the angle β as the angle between the laser beam and the line connecting A and B in the point that is further away from the scanner (in our example that is A). In practice, the angle β turns out to be a valuable piece of information to determine if the points A and B lie on the same object or not.

Given the nature of the laser range measurements, we know the distance $\|OA\|$ as it corresponds to the first laser measurement as well as $\|OB\|$ (second laser measurement). We will call these range measurements d_1 and d_2, respectively. One can use this information to calculate β by applying trigonometric equations

$$\beta = \text{atan2}(\|BH\|, \|HA\|) = \text{atan2}(d_2 \sin\psi, d_1 - d_2 \cos\psi),$$

where ψ is the known angle between the beams and is usually provided in the documentation of the scanner. The right image in Fig. 5 illustrates the computation in the xy-plane from a top-down view of the scene. Note that we can compute the angle β for pairs of points A and B that are neighbours either in row or in column direction in the range image. In the first case, the angle ψ corresponds to the angular increment in row direction, in the other case to the increment in column direction.

The intuition behind the angle β is that it stays relatively large for most objects and only takes small values if the depth difference between neighbouring points given the range image is substantially larger than their displacement in the image plane that is defined through the angular resolution of the scanner. This insight allows us to define a parameter θ that acts as a threshold on the angle β. This threshold enables us to take a decision about whether to separate any two points in the range image into separate clusters or merge them into one. If β is smaller than the user-defined value θ, we argue that the change in depth is too large and take the decision

to separate the points into different segments. Otherwise, the points are considered as lying on the same object.

A threshold-based criterion on β is clearly a heuristic but works well in practice as we will illustrate in the experimental evaluation. A failure case can be a situation in which the scanned object is planar, such as a wall, and oriented nearly parallel to the laser beams. In this case, the angle β will be small and it is, therefore, likely for the object to be split up into multiple segments. This essentially means that if β is smaller than θ, it is difficult to find out if two points originate on two different objects or just lie on a planar object nearly parallel to the beam direction. However, despite this shortcoming, our experiments suggest that the method is still useful in practice. The aforementioned behaviour occurs rarely and if so, it usually results only in an over-segmentation of particularly inclined planar objects.

With the separating threshold in mind, we approach the segmentation directly in the range image. We regard two endpoints as being neighbours stemming from the same object if they are neighbours in a the range image (we use an N4 neighbourhood on the grid) and the angle β between them is larger than θ. Given this definition of a neighbourhood, we can view the segmentation problem as the problem of finding the connected 2D components exploiting the structure of the range image and the constraint on β.

Algorithm 2 Range Image Labelling

```
1:  procedure LabelRangeImage(R)
2:      Label ← 1, L ← zeros(R_rows × R_cols)
3:      for r = 1 ... R_rows do
4:          for c = 1 ... R_cols do
5:              if L(r, c) = 0 then
6:                  LabelComponentBFS(r, c, Label);
7:                  Label ← Label + 1;
8:  procedure LabelComponentBFS(r, c, Label)
9:      queue.push({r, c})
10:     while queue is not empty do
11:         {r, c} ← queue.top()
12:         L(r, c) ← Label
13:         for {r_n, c_n} ∈ Neighbourhood{r, c} do
14:             d_1 ← max(R(r, c), R(r_n, c_n))
15:             d_2 ← min(R(r, c), R(r_n, c_n))
16:             if atan2 (d_2 sin ψ)/(d_1 − d_2 cos ψ) > θ then
17:                 queue.push({r_n, c_n})
18:         queue.pop()
```

Algorithm 2 depicts the algorithm that we use to find the connected components that define the segments. We use a variant of a pass-through filter with complexity $\mathcal{O}(N)$, where N is the number of pixels, i.e. the number of range readings per scan. The algorithm guarantees visiting each point in the range image at maximum twice. Please note that at this point in time all pixels of the range image that were labelled as ground (see Sect. 3) are set to zero and do not take part in the following procedure.

We start in the top left corner of the range image and pass through every pixel from top to bottom, left to right (lines 4–5). Whenever we encounter an unlabelled pixel (line 6), we start a breadth-first search from this pixel (line 7). The goal of this search is to label every pixel of this component. For this purpose, the BFS uses a queue (lines 10–12) and an N4 neighbourhood consisting of the left, right, lower and top pixels (line 14). The decision if a point in the N4 neighbourhood should be added to the queue of the BFS is taken based on the angle β generated by the neighbour and the current point (lines 15–18). This procedure guarantees that the whole connected component will receive the same label. Once the queue of BFS is empty, we continue to traverse the range image sequentially until we reach a new unlabelled point.

It has to be noted that the connected components algorithm is not the main contribution of this work but its effective application to the segmentation of range images considering the value of β for two neighbouring measurements. For more information on the comparison between different implementations of connected components algorithms, we refer the reader to Cabaret et al. (2014). Overall, our approach yields an easy-to-implement and fast method that does not require a lot of parameters tuning to achieve good segmentation performance.

5 Experimental Evaluation

Our experiments are designed to show the capabilities of our method and to support our key claims, which are: (1) all computations can be executed fast, even on a single core of a mobile CPU with around 70 Hz, (2) we can segment typical 3D range data obtained by mobile robots into meaningful segments, and (3) the approach performs well on sparse data such as those obtained from a 16-beam Velodyne Puck scanner. In our evaluation, we also provide comparisons to the grid-based segmentation method proposed by Teichman and Thrun (2012) as used by Behley et al. (2013) as well as to Euclidean clustering implemented in the point cloud library PCL. Throughout all experiments, we used our default parameter $\theta = 10°$.

5.1 Runtime

The first experiment is designed to support the claim that our approach can be executed fast supporting online processing in real time. We, therefore, tested our approach on point clouds computed with different Velodyne laser scanners and processed the data on different computers. On the robot, we used an Acer notebook with an i5 5200U 2.2 GHz CPU and we also processed the data on a desktop computer with an i7 4770K 3.5 GHz CPU, in both cases using only one core of the CPU.

Table 1 summarizes the runtime results for nearly 2,500 point clouds (generated by a single revolution of the scan-

Table 1 Average runtime and std. dev. per 360° laser scan

Scanner	Segmentation only		Ground removal + segmentation	
	Mobile	Desktop	Mobile	Desktop
	i5 U5200 2.2 GHz	i7 4770K, 3.5 GHz	i5 U5200 2.2 GHz	i7 4770K 3.5 GHz
64-beam	8.6 ± 2.6 ms	4.7 ± 1.2 ms	13.3 ± 1.0 ms	8.6 ± 0.6 ms
	116 Hz	212 Hz	74 Hz	116 Hz
32-beam	4.4 ± 1.2 ms	2.6 ± 0.5 ms	8.3 ± 0.7 ms	4.5 ± 0.7 ms
	227 Hz	385 Hz	120 Hz	222 Hz
16-beam	2.4 ± 0.5 ms	1.5 ± 0.2 ms	4.0 ± 0.8 ms	2.8 ± 1.0 ms
	416 Hz	667 Hz	250 Hz	354 Hz

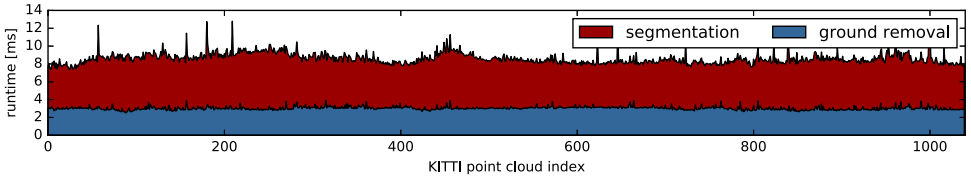

Fig. 6 Timings obtained on the KITTI dataset. The *x*-axis depicts the index of individual point clouds while the *y*-axis shows the processing time in ms

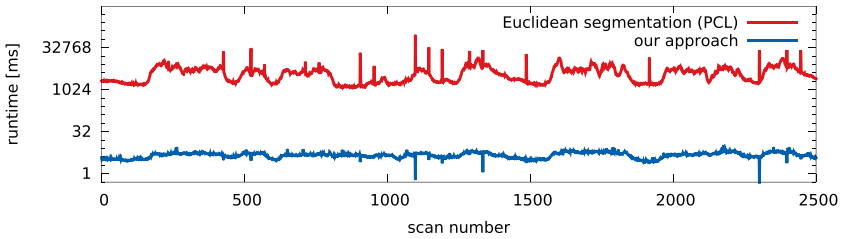

Fig. 7 Timings for segmenting approximately 2500 scans from a 64-beam Velodyne dataset with our approach and Euclidean segmentation from PCL (without ground removal)

ner) recorded in urban outdoor environments. The numbers support our first claim, namely that the computations can be executed fast and in an online fashion. The frame rate of our segmentation pipeline including ground removal is multiple times faster than the frame rate of the sensor. On a mobile i5 CPU, we achieve average frame rates of 74–250 Hz for the whole approach and 116–354 Hz on an i7 computer. The pure segmentation without ground removal can run with up to 667 Hz. We obtained similar timings on the publicly available KITTI datasets by Geiger et al. (2013), see Fig. 6.

We also compared the speed of our segmentation pipeline to Euclidean clustering for segmentation as provided by PCL. For a fair comparison, we used the same ground removal for both approaches and thus the reported timing refers to the segmentation only. As can be seen from Fig. 7, our approach is on average around 1000 times faster than Euclidean clustering in the 3D space, here using 64-beam Velodyne data.

5.2 Segmentation Results

The next set of experiments is designed to illustrate the obtained segmentation results. We consider the results on 16- and 64-beam laser range data. For the 64-beam evaluation, we rely on the publicly available street scenes dataset by Moosmann (2013) and the KITTI dataset by Geiger et al. (2013), while we recorded the 16-beam datasets using our robot in Bonn, Germany, see also Fig. 1.

We evaluate the performance of our method and compare it to a popular grid-based approach by Behley et al. (2013) and to segmentation through Euclidean clustering as provided by PCL. For that purpose, we manually segmented 30 point clouds from different scenes and ran all three methods on these data varying their parameters. For our method, we have chosen different values for θ, while for the grid-based approach we have varied the size of the grid cells. We have chosen values for θ from 5° to 45° and for the grid cell resolution (grid-based) and the distance threshold (Euclidean) values between 0.05 and 1.25 m. We have evaluated the performance of the algorithms by counting how many of the manually labelled objects have been found by the algorithms. For every ground truth cluster, we search for a found segment with the biggest overlap. We consider the cluster as correctly found if the point-wise overlap is substantial (80% in our implementation). We then count the number of successful matches and divide them by the number of expected ground truth clusters. We compute the performance measure for every scan and present the mean and standard deviation of these values with relation to the chosen parameter in Fig. 8. As can be seen with $\theta = 10°$, our method outperforms the grid-based approach in terms of segmentation quality in all parameters settings. In comparison to Euclidean clustering, our approach shows a comparable performance on the 64-beam datasets, while being around three orders of magnitudes faster (4 ms vs. 4 s per scan). This nicely illustrates the ben-

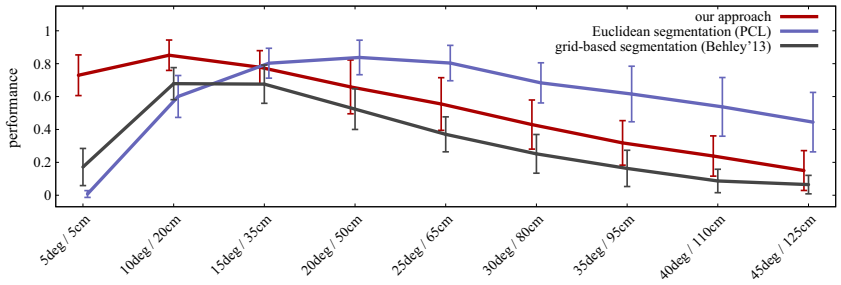

Fig. 8 Performance of our algorithm computed as a fraction of the number of found objects over the number of all manually labelled objects in the scene compared to the grid-based segmentation by Behley et al. (2013) and segmentation through Euclidean clustering as provided by PCL for varying parameters on 30 different, manually labelled outdoor 3D data. On the x-axis, the first value is the parameter θ for our method and the second one serves as both the cell size for the grid-based approach and as the distance threshold for the Euclidean clustering approach

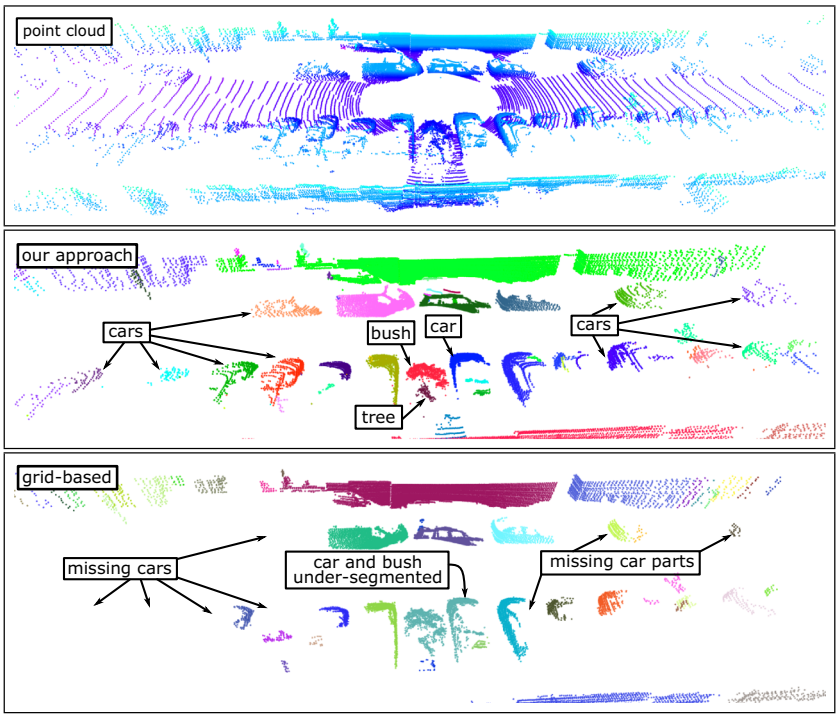

Fig. 9 *Top* Point cloud of an outdoor scene taken with a 64-beam Velodyne (shown for illustration only). *Middle* Our segmentation that provides correct segmentation even for distant objects while not under-segmenting the close ones. *Bottom* Segmentation provided by a grid-based approach with cell size set to 0.2. There is a number of cars that are situated further from the sensor missing and one car is merged with a bush

efits of our method for online processing. Typical examples of a segmentation are shown in Figs. 9 and 10, both using a 64-beam Velodyne scanner.

Finally, we aim at supporting our claim that our segmentation pipeline can handle sparse data coming from a scanner with 16 beams in the vertical direction (Velodyne VLP-16) well. For this purpose, we analysed the results using data recorded from our scanner and compared them to manually labelled ground truth clouds. Examples are depicted in Fig. 11. Although this is only a qualitative evaluation, we can clearly see that our approach handles the sparse range data better than the approaches that work in the space of 3D points.

In summary, our evaluation suggests that our method provides competitive segmentation results compared to existing methods on dense range images and outperforms them on sparse scans. At the same time, our method is fast enough for online processing and has small computational demands. Thus, we supported all our claims with this experimental evaluation.

6 Conclusion

This paper presents a fast and easy-to-implement method for 3D laser range data segmentation including fast ground removal. Instead of operating in the 3D space, our approach performs all computations directly on the range images. This speeds up the segmentation of the individual range images and allows us to directly exploit neighbourhood rela-

Fig. 10 An example segmentation of a group of people from KITTI dataset

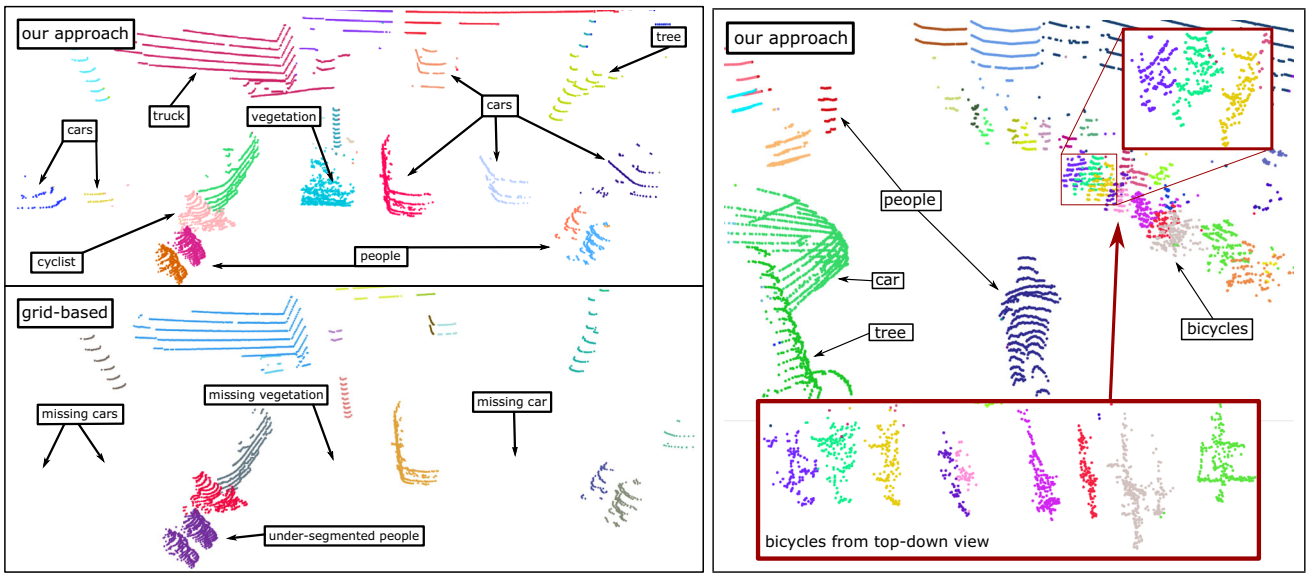

Fig. 11 *Left top* Our segmentation of an example outdoor scene taken with a 16-beam Velodyne. Our approach was able to find objects omitted by the grid-based method while correctly segmenting people that stand close to each other. *Left bottom* Grid-based segmentation result. Some objects are missing and people on the bottom left are under-segmented. *Right* An outdoor scene recorded with a 16 beam Velodyne that shows that our approach is able to segment even complicated scenes with multiple small objects like bicycles placed very close to each other. The grid-based approach in this scene merged all the bicycles into two big clusters. The images are omitted for space reasons

tions. It enables us to successfully segment even sparse laser scans like those recorded with a 16-beam Velodyne scanner. We implemented and evaluated our approach on different publicly available and self-recorded datasets and provide comparisons to other existing techniques. On a single core of a mobile i5 CPU, we obtain segmentation results at average frame rates between 74 and 250 Hz and can run up to 667 Hz on an i7 CPU. We will release our code that can either be used standalone with C++ or as a ROS module.

Acknowledgements We thank JENS BEHLEY for fruitful discussions and for providing his implementation of grid-based segmentation. Further thanks to FRANK MOOSMANN for sharing his data.

References

Abdullah S, Awrangjeb M, Lu G (2014) LiDAR segmentation using suitable seed points for 3d building extraction. Int Arch Photogramm Remote Sens Spatial Inf Sci 40(3):1–8

Bansal M, Matei B, Sawhney H, Jung SH, Eledath J (2009) Pedestrian detection with depth-guided structure labeling. In: International Conference on Computer Vision Workshops, pp 31–38

Behley J, Steinhage V, Cremers AB (2013) Laser-based segment classification using a mixture of bag-of-words. In: International Conference on Intelligent Robots and Systems, pp 4195–4200

Bogoslavskyi I, Stachniss C, (2016) Fast range image-based segmentation of sparse 3d laser scans for online operation. In: International Conference on Intelligent Robots and Systems

Cabaret L, Lacassagne L, Oudni L (2014) A review of world's fastest connected component labeling algorithms: speed and energy estimation. In: International Conference on Design and Architectures for Signal and Image Processing, pp 1–6

Choe Y, Ahn S, Chung MJ (2012) Fast point cloud segmentation for an intelligent vehicle using sweeping 2d laser scanners. In: International Conference on Ubiquitous Robots and Ambient Intelligence, pp 38–43

Dewan A, Caselitz T, Tipaldi G, Burgard W (2016) Motion-based detection and tracking in 3d lidar scans. In: IEEE International Conference on Robotics & Automation

Douillard B, Underwood J, Kuntz N, Vlaskine V, Quadros A, Morton P, Frenkel A (2011) On the segmentation of 3d lidar point clouds. In: IEEE International Conference on Robotics & Automation. IEEE, pp 2798–2805

Douillard B, Underwood J, Vlaskine V, Quadros A, Singh S (2014) A pipeline for the segmentation and classification of 3d point clouds. International Symposium on Experimental Robotics. Springer, New York, pp 585–600

Endres F, Plagemann C, Stachniss C, Burgard W (2009) Unsupervised discovery of object classes from range data using latent dirichlet allocation. Robot Sci Syst 2:113–120

Floros G, Leibe B (2012) Joint 2d-3d temporally consistent semantic segmentation of street scenes. In: IEEE Conference on Computer Vision and Pattern Recognition, pp 2823–2830

Geiger A, Lenz P, Stiller C, Urtasun R (2013) Vision meets robotics: The KITTI dataset. Int J Robot Res 32(11):1231–1237

Golovinskiy A, Funkhouser T (2009) Min-cut based segmentation of point clouds. In: International Conference on Computer Vision Workshops, pp 39–46

Gorte B, Oude Elberink S, Sirmacek B, Wang J (2015) Tree separation and classification in mobile mapping lidar data. Int Arch Photogramm Remote Sens Spatial Inf Sci 40(3/W3):607–612

Hackel T, Wegner JD, Schindler K (2016) Fast semantic segmentation of 3d point clouds with strongly varying density. ISPRS Ann Photogramm Remote Sens Spatial Inf Sci 3:177–184

Hanel A, Klöden H, Hoegner L, Stilla U (2015) Image based recognition of dynamic traffic situations by evaluating the exterior surrounding and interior space of vehicles. Int Arch Photogramm Remote Sens Spatial Inf Sci 40(3):161–168

Hebel M, Stilla U (2008) Pre-classification of points and segmentation of urban objects by scan line analysis of airborne lidar data. Int Arch Photogramm Remote Sens Spatial Inf Sci 37(B3a):105–110

Hermans A, Floros G, Leibe B (2014) Dense 3d semantic mapping of indoor scenes from rgb-d images. In: IEEE International Conference on Robotics & Automation, pp 2631–2638

Himmelsbach M, Hundelshausen FV, Wuensche H (2010) Fast segmentation of 3d point clouds for ground vehicles. IEEE Intelligent Vehicles Symposium, pp 560–565

Klasing K, Wollherr D, Buss M (2008) A clustering method for efficient segmentation of 3d laser data. IEEE International Conference on Robotics & Automation, pp 4043–4048

Korchev D, Cheng S, Owechko Y, Kim K (2013) On real-time lidar data segmentation and classification. Int Conf Image Process Comput Vision Pattern Recogn 1:42–49

Kümmerle R, Ruhnke M, Steder B, Stachniss C, Burgard W (2013) A navigation system for robots operating in crowded urban environments. In: IEEE International Conference on Robotics & Automation, pp 3225–3232

Leibe B, Schindler K, Cornelis N, Van Gool L (2008) Coupled object detection and tracking from static cameras and moving vehicles. IEEE Trans Pattern Anal Mach Intell 30(10):1683–1698

Leonard J, How J, Teller S, Berger M, Campbell S, Fiore G, Fletcher L, Frazzoli E, Huang A, Karaman S, Koch O, Kuwata Y, Moore D, Olson E, Peters S, Teo J, Truax R, Walter M, Barrett D, Epstein A, Maheloni K, Moyer K, Jones T, Buckley R, Antone M, Galejs R, Krishnamurthy S, Williams J (2008) A perception-driven autonomous urban vehicle. J Field Robot 25(10):727–774

Menze M, Heipke C, Geiger A (2015) Joint 3d estimation of vehicles and scene flow. ISPRS Ann Photogramm Remote Sens Spatial Inf Sci **2**(3/W5):427–434

Moosmann F (2013) Interlacing self-localization, moving object tracking and mapping for 3d range sensors, Ph.D. thesis. KIT

Moosmann F, Pink O, Stiller C (2009) Segmentation of 3d lidar data in non-flat urban environments using a local convexity criterion. In: Intelligent Vehicles Symposium, pp 215–220

Ošep A, Hermans A, Engelmann F, Klostermann D, Mathias M, Leibe B (2016) Multi-scale object candidates for generic object tracking in street scenes. In: IEEE International Conference on Robotics & Automation, pp 3180–3187

Petrovskaya A, Thrun S (2008) Model based vehicle tracking for autonomous driving in urban environments. Robot Sci Syst. 34. http://www.roboticsproceedings.org/rss04/p23.pdf

Pylvanainen T, Roimela K, Vedantham R, Itaranta J, Grzeszczuk R (2010) Automatic alignment and multi-view segmentation of street view data using 3d shape priors. In: Symposium on 3D Data Processing. Visualization and Transmission, vol 737, pp 738–739

Savitzky A, Golay MJ (1964) Smoothing and differentiation of data by simplified least squares procedures. Anal Chem 36(8):1627–1639

Steinhauser D, Ruepp O, Burschka D (2008) Motion segmentation and scene classification from 3d lidar data. In: Intelligent Vehicles Symposium, pp 398–403

Strom J, Richardson A, Olson E (2010) Graph-based segmentation for colored 3d laser point clouds. International Conference on Intelligent Robots and Systems, pp 2131–2136

Teichman A, Thrun S (2012) Tracking-based semi-supervised learning. Int J Robot Res 31(7):804–818

Velizhev A, Shapovalov R, Schindler K (2012) Implicit shape models for object detection in 3d point clouds. ISPRS Ann Photogramm Remote Sens Spatial Inf Sci 1(3):179–184

Wang J, Shan J (2009) Segmentation of lidar point clouds for building extraction. In: Annual Conference of the American Society for Photogrammetry and Remote Sensing, pp 9–13

Weinmann M, Jutzi B (2015) Geometric point quality assessment for the automated, markerless and robust registration of unordered tls point clouds. ISPRS Ann Photogramm Remote Sens Spatial Inf Sci 2(3/W5):89–96

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

Wurm KM, Stachniss C, Burgard W (2008) Coordinated multi-robot exploration using a segmentation of the environment. International Conference on Intelligent Robots and Systems, pp 1160–1165