CNN for Very Fast Ground Segmentation in Velodyne LiDAR Data

Martin Velas, Michal Spanel, Michal Hradis, and Adam Herout

***Abstract*— This paper presents a novel method for *ground segmentation in Velodyne* point clouds. We propose an encoding of sparse 3D data from the Velodyne sensor suitable for training a *convolutional neural network (CNN)*. This general purpose approach is used for segmentation of the sparse point cloud into ground and non-ground points. The LiDAR data are represented as a multi-channel 2D signal where the horizontal axis corresponds to the rotation angle and the vertical axis represents channels – laser beams. Multiple topologies of relatively shallow CNNs (i.e. 3-5 convolutional layers) are trained and evaluated, using a manually annotated dataset we prepared. The results show significant improvement of performance over the state-of-the-art method by Zhang et al. in terms of *speed* and also minor improvements in terms of accuracy.**

I. Introduction

Recent development in exploration and 3D mapping of the environment surrounding a mobile robot aims at techniques which capture semantic information besides the simple geometrical properties. The analysis of scene dynamics was successfully used in the task of object detection (pedestrians, cars, bicycles, ...) [2], and by filtering out moving objects, 3D maps capturing only static parts of the environment can be built [3]. Such maps are useful for the localization or the motion planning where measurements of moving objects are undesirable and introduce motion artifacts into the map. Successful methods for the *detection and tracking of moving objects (DATMO)* assume that the way, in which sensors are used, causes that only the objects (static or dynamic) are captured [4], or that the ground can be detected (see Fig. 1) and filtered out in the preprocessing stage [5]–[9]. For these purposes, we intend to reliably and efficiently *segment the data to ground/non-ground* parts. We consider the ground to be every surface traversable by commonly moving objects (pedestrians, cars, bikes, etc.).

In these DATMO systems, the ground detection is typically based on primitive features with low discriminative capabilities. The state of the art technique for robust ground segmentation by Zhang et al. [1] achieves good results in terms of accuracy by building a Markov Random Field (MRF) and inference using the Loopy belief propagation. Unfortunately, the robustness of this method is achieved by compromising its time efficiency (over 2 minutes per frame).

All authors are with with the Faculty of Information Technology, Brno University of Technology, Czech Republic {ivelas|spanel|ihradis|herout} at fit.vutbr.cz

*This work has been supported by the Artemis JU grant agreement ALMARVI (no. 621439), the TACR Competence Centres project V3C (no. TE01020415), and the IT4IXS IT4Innovations Excellence project (LQ1602).

**We would like to thank Mingfang Zhang for sharing her source code [1], making evaluation of both approaches feasible.

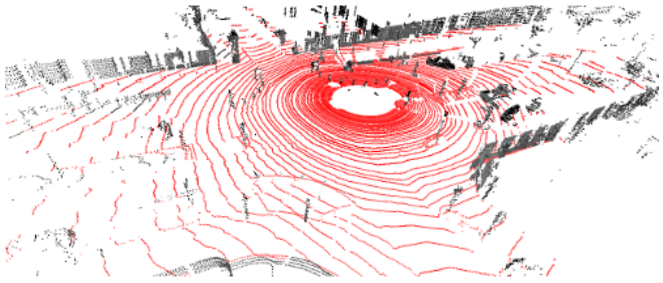

Fig. 1: Expected segmentation of the Velodyne LiDAR point cloud into sets of ground (red) and non-ground (grey) points.

The *Velodyne* sensor – nowadays common source of *LiDAR* (Light Detection And Ranging) data – captures the full 3D information about environment, in contrast to simple range finders, providing information about occupancy in a certain height around the robotic platform only. Currently, the most powerful model HDL-64E covers full 360° horizontal field and 26.8° vertical field of view, and with up to 15 Hz frame rate, captures over 1.3 M of points per second. This sensor scans the surrounding area by 64 rotating laser beams while each beam produces one *ring* of 3D points (red circles in Fig. 1).

Since the breakthrough in machine learning after introduction of AlexNet [10], the attractiveness of *Convolutional Neural Networks (CNNs)* has grown rapidly and this model was successfully used for many computer vision tasks including image classification, object detection, face recognition, semantic segmentation [11], etc. In this work, we deployed convolutional neural networks for the task of *ground segmentation* in sparse Velodyne point cloud data. We designed multiple networks with shallow topologies (3-5 convolutional layers) fulfilling the requirements for robustness and accuracy. We trained and evaluated them by using a hand-annotated dataset.

The main contributions of this work are the following:

- we show that the *sparse 3D LiDAR data can be encoded into a multi-channel 2D signal* (analogous to HHA encoded range images [12] or LiDAR data encoding in vehicle detection task [13]) and processed by convolution neural network;
- new *approach to ground segmentation in Velodyne point clouds using CNN* which outperforms current state of the art in accuracy and time performance.

Besides this, we developed a *semi-automatic ground annotation* tool and we annotated a part of the KITTI tracking dataset. Source code of the annotation tool and LiDAR point clouds preprocessing methods, design and configuration of the trained convolutional networks, as well as annotated

Preprint submitted to 2018 IEEE International Conference on Robotics and Automation. Received September 14, 2017.

ground truth data are publicly available[1].

II. Related work

As mentioned above, we define the ground as a surface traversable by commonly moving objects. A similar definition has been already used for an outdoor robot [14]. The traversability estimation was performed using geometric features (extracted from stereo-vision) and texture features (from RGB images). By clustering, the labels are assigned to parts of the surrounding environment. Compared to our approach, this method requires explicit feature specification and different type of input data – stereo RGB vision, IMU, and motor current sensor.

Convolutional networks were deployed for learning rich descriptors of RGB-D data [12] useful for per-pixel object detection. The input of networks encodes horizontal disparity (equivalent to the range), height and normals angle. Our work proposes a similar type of encoding suitable for processing the sparse LiDAR data. Since the normals can not be robustly estimated in these data, the angles are not used.

Many DATMO (detection and tracking of moving objects) methods segment and filter out the ground measurements from LiDAR data in a preprocessing stage [5]–[9], [15], [16]. These approaches usually rely on primitive features with low discriminative capabilities like mean or variance of measured height in a certain small area, or changes in the elevation between the rings in Velodyne data.

More traditional DATMO methods operate over data from simple laser rangefinders [4], assuming the measurements provided by LiDAR positioned approximately parallel to the ground surface, capturing only the upright (moving and/or static) objects and not the ground. Over such data, the occupancy grid can be built and detection of movement is performed by particle filtering.

When data from multiple laser sensors including Velodyne 3D LiDAR are fused [9], building the occupancy grid starts to be an issue, since the sensors cover a significantly larger area including the ground. The ground measurements must be recognized and filtered out in order to build a valid occupancy grid representing free space, the space occupied by obstacles, and currently unobserved areas. For sake of effectivity, authors [9] selected a computationally inexpensive approach where all measurements within a certain height range are considered to be ground. Besides the sensitivity to selection of optimal thresholds, the robustness/repeatability of such approach is far from the optimal (see Fig. 2).

The motion detection generalized to motion field estimation [5] in a polar grid benefits from the large area covered by the Velodyne LiDAR scanner. The preprocessing step, same as in the previously mentioned work – i.e. the ground detection and filtering – is performed as well. Using the simple thresholding, this method shares the same disadvantages. The areas (polar grid cells), fulfilling at least one of the following conditions, are considered to be ground: the average height fits an exactly defined range, the standard deviation of the height is below a certain threshold, or the difference between the minimal and the maximal height inside the cell is below another threshold. A very similar approach with only small modifications was used by Asvadi et al. [8] in a DATMO system operating over a regular orthogonal grid. The area within one grid cell is considered to be ground if both the mean height and the standard deviation of the heights fit below a predefined threshold.

[1] https://www.github.com/to-be-added-after-acceptance

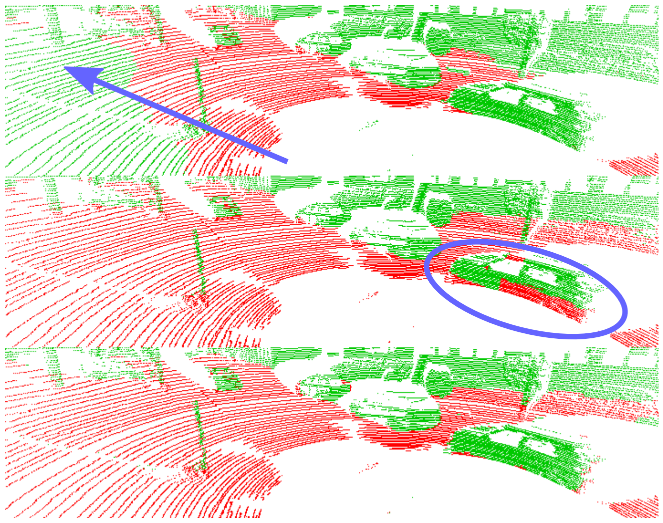
Fig. 2: Different results of ground segmentation methods. **top:** Simple height thresholding can not deal with terrain elevation; **middle:** Loopy belief propagation [1] produces incorrect results when objects are close to the sensor; **bottom:** our method.

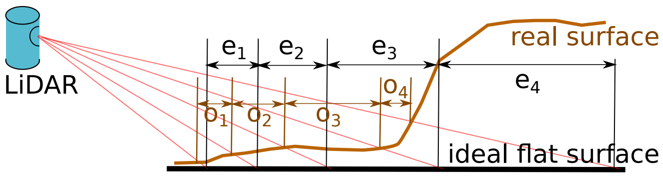

Fig. 3: Ground detection by comparison of expected range difference e_i with observed difference o_i. Since $e_4 - o_4 > th$, the border between the obstacle and the ground is found [6].

Other approaches analyse changes in the elevation in order to segment the ground in Velodyne LiDAR scans [7], [15], [16]: each vertical slice consisting of all points captured at exactly the same moment by all laser rays, is analysed separately. Three points A, B, C from adjacent rings form two vectors $\overrightarrow{AB}$ and $\overrightarrow{BC}$. If the dot product of these (normalized) vectors is above a certain threshold, a significant change of elevation – the breakpoint – is found. Such breakpoints form the border between the ground (points between the sensor and the breakpoint) and an obstacle (points behind the breakpoint). Besides the lack of robustness, this approach does not allow to reason about the space behind the first obstacle where the ground can be observed again.

Analysis of ranges differences between two adjacent Velodyne rings (Fig. 3) was also used for the ground segmentation [6] in LiDAR data. On the ideal flat horizontal surface, the expected range difference e_i between two adjacent rings can be computed, assuming the height and the vertical angle of each laser beam is known. This range difference decreases with increasing elevation of the surface. At the ideal vertical obstacle, this difference becomes zero.

Besides the previously mentioned DATMO methods, the ground detection and filtering plays important role in point cloud registration by scan segments matching [17]. However, in a preprocessing step, the ground points are also detected by thresholding the mean and the variance of vertical height withing the cells of voxel grid [18].

The lack of accuracy and robustness in previously mentioned methods, mostly caused by the fixed thresholding of simple features with low discriminative power, was overcome by the inference in *Markov Random Field (MRF)* [1]. Although the introduced 3D volumetric grid is built by estimation of a slope in each vertical slice in a similar way to 2D occupancy grids, the final segmentation to ground/obstacle is not made directly. At first, based on the slope detected, the points are categorized as unknown, probably ground, probably obstacle, and probably obstacle borders. This categorization implies the initial cost assigned to each volumetric element of the regular 3D polar grid. The key improvement is done by Loopy Belief Propagation inference in order to estimate ground height within a certain region. All measurements within this region with a smaller height are considered to be the ground points. The rest is classified as non-ground. Unfortunately, the robustness of this method is achieved by compromising its time efficiency. In our experiments with the original MATLAB implementation, kindly provided by the authors, the processing of a single Velodyne HDL-64E frame takes approximately 145 seconds.

The key improvement achieved by our method is the reduction of time complexity of the ground segmentation process to fraction of the time required by Loopy Belief Propagation [1], while slightly better results in terms of accuracy were achieved as well. Processing of a single Velodyne frame by our *L05+deconv* network takes 140 ms on average, using only CPU. By using GPU (GeForce GTX 770), the processing time is further reduced to 7 ms per frame.

Simultaneously with our work, the Baidu research team [13] proposed a similar encoding of sparse LiDAR data into 2D matrices for the vehicles detection by convolutional neural networks. Our encoding differs from their work in polar bin aggregation of LiDAR points (described in Sec. III-A) to improve the stability of prediction.

III. Proposed Ground Segmentation Method

The goal of our method is to assign a *binary label ground/non-ground* (1) to each 3D point $\boldsymbol{p} \in \boldsymbol{P}$ measured by the LiDAR sensor. The point cloud elements $\boldsymbol{p}$ are represented by *3D coordinates* originating at the LiDAR sensor position, accompanied by the laser *intensity* reading and the *ring* ID identifying the source laser beam which was used to measure the point $\boldsymbol{p} = [p_x, p_y, p_z, p_i, p_r]$. Since

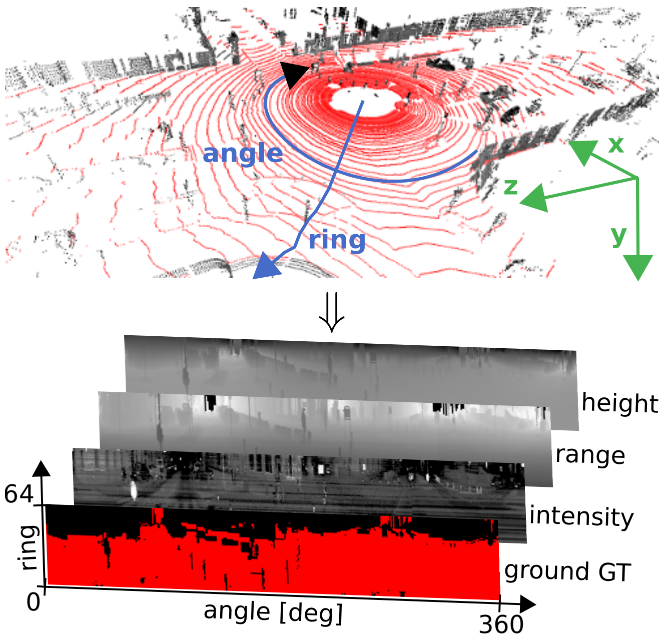

Fig. 4: Transformation of the sparse Velodyne point cloud into the multi-channel dense matrix. Each row represents measurements of a single laser beam done during one rotation of the sensor. Each column contains measurements of all 64 laser beams captured at a specific rotational angle at the same time.

we do not assign the ground label to each LiDAR point separately, we solve the assignment (2) of binary labels to all the points jointly.

$$g : \boldsymbol{P} \to \{0,1\} \tag{1}$$

$$G : \mathbb{P} \to \{0,1\}^{|\boldsymbol{P}|}, \quad \boldsymbol{P} \in \mathbb{P} \tag{2}$$

A. Encoding Sparse 3D Data Into a Dense 2D Matrix

In order to process the Velodyne LiDAR data by a convolutional neural network, we *encode* (3) the original *sparse point cloud* $\boldsymbol{P}$ into a multi-channel *dense matrix* $\boldsymbol{M}$. The original 3D data are treated as a 2D signal in the domain of the ring (the ID of the source laser beam) and the horizontal angle, as illustrated in Fig. 4. The size of the resulting matrix $\boldsymbol{M}$ depends on the number of rings in the LiDAR frame (i.e. number of laser beams used) and the sampling rate R of the horizontal angle. In our experiments, we used Velodyne LiDAR HDL-64E with 64 rays and resolution $R = 1°$.

$$G(\boldsymbol{P}) = \tilde{G}(\boldsymbol{M}), \quad \boldsymbol{M} = \mathcal{E}(\boldsymbol{P}) \tag{3}$$

At first, the point cloud is aggregated into the polar bins $\boldsymbol{b_{r,c}}$ (6) analogous to our previous work [19]. All the points assigned to the same bin share the same ring ID r (points captured by the same laser beam) and fit into the same polar cone $c = \varphi(\boldsymbol{p})$ (7), computed according to the horizontal angle of the point. Each polar bin is encoded into the element $\mathbf{m_{r,c}}$ of matrix $\boldsymbol{M}$ in its r-th row and c-th column (4). Since multiple points fall into the same bin (the horizontal representation of our encoding is coarser than original Velodyne resolution), a single representative

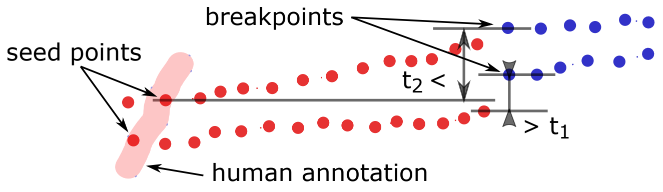

Fig. 5: Flooding the human made annotation from seed points along the ring. The ground points are red. When the breakpoint is found (first of blue not-ground points), the flooding is stopped.

of the bin is found as the average (5). Moreover, since the horizontal index in the matrix $\boldsymbol{M}$ encodes the rotational angle in the 3D horizontal XZ plane, we can reduce the number of channels by replacing XZ coordinates p_x, p_z by depth (or range) value $d = \|p_x, p_z\|_2$ without the loss of any information.

$$\boldsymbol{m}_{r,c} = \varepsilon(\boldsymbol{b}_{r,c}) \tag{4}$$

$$\varepsilon(\boldsymbol{b_{r,c}}) = \frac{\sum\limits_{\boldsymbol{p}\in\boldsymbol{b_{r,c}}} [p_y, \|p_x, p_z\|_2, p_i]}{|\boldsymbol{b_{r,c}}|} \tag{5}$$

$$\boldsymbol{b_{r,c}} = \{p \in \boldsymbol{P} \,|\, p_r = r \wedge \varphi(\boldsymbol{p}) = c\} \tag{6}$$

$$\varphi(\boldsymbol{p}) = \left\lfloor \frac{\text{atan}(\frac{p_z}{p_x}) + 180°}{\frac{360°}{R}} \right\rfloor \tag{7}$$

In case of empty bins (e.g. no measurement exists in this area due to the sensor limits), the value in the matrix $\boldsymbol{M}$ is linearly interpolated from the neighbourhood.

B. Training Dataset

The most serious issue in development of the proposed system was the lack of training data, especially missing annotations of ground data in the Velodyne scans. The developement of KITTI Semantic Segmentation dataset[2] is still in progress and only small subsets are available at the moment. The only annotations relevant to our task were created by Richard Zhang [20] in his work on semantic segmentation of urban scenes. However, Zhang used the LiDAR point clouds as a supplementary data only, and annotations were made for RGB camera images in the first place. These annotations were probably back-projected into the LiDAR frames and spread across consequent frames which caused serious inaccuracies in the ground annotations and made these data unsuitable for our training and testing.

Therefore we prepared a *semiautomatic tool for ground annotation* in 3D Velodyne data[3]. Using a pen-like drawing tool, the user highlights certain ground points as ground seed points $\boldsymbol{p}^s$. From these points, the annotation *automatically floods* along the ring until a breakpoint $\boldsymbol{p}^b$ is found (see Fig. 5). The breakpoint is defined as the first point, where the height difference with respect to the previous point $|p_y^b - p_y^{i-1}| > t_1$, or with respect to the seed point $|p_y^b - p_y^s| > t_2$, is above a respective threshold. When annotating the dataset, we found the values $t_1 = 3\,\text{cm}$ and $t_2 = 7\,\text{cm}$ work best as they save annotator's time and they reduce manual changes.

[2] http://www.cvlibs.net/datasets/kitti/eval_semantics.php
[3] https://www.github.com/to-be-added-after-acceptance

Using this tool, we prepared *accurate annotations of the ground* in 3D LiDAR data for a subset of KITTI Tracking Dataset – the same data as was annotated by [20] in RGB images. The subset consists of 8 data sequences taken at different places of urban and suburban environments. In total, there are 252 frames captured in $1\,\text{s}$ interval. We randomly split those frames into training and evaluation set in $70 : 30$ ratio.

Since the amount of available annotated data is quite small, we prepared automatic artificial annotations for the rest of the KITTI Tracking Dataset ($19\,\text{k}$ frames) by thresholding simple features, like the mean and the variance of height, and the distance and the elevation differences between rings, as used in the previous works [5]–[9], [15], [16]. These artificial annotations are used for CNNs pretraining. The resulting parameters are used as initial weights of convolutional kernels for further training on more precise human annotations.

We also tried to use data augmentation and generate artificial 3D LiDAR frames automatically. Unfortunately, this approach proved to be infeasible, since the available 3D models are not detailed enough, lack fine surface details, and substitute this structure information (trees, bushes, curbs, etc.) by texturing flat surfaces (so called billboarding).

C. Topology and Training of the Proposed Networks

Because of the small amount of annotated training data, we used *shallow CNN architectures* only. All the networks are fully convolutional. They consists of convolution and deconvolution layers with ReLU non-linearities. Gradient descent is used as the optimization method for the training. The most interesting and successful topologies we experimented with are presented in Fig. 6.

The multi-channel matrix $\boldsymbol{M}$, obtained by the encoding described in III-A, is the input of all proposed networks. The probability of being ground point $p_g = p(g(\boldsymbol{p}) = 1)$ is estimated for each pixel of this matrix. Therefore, the output of all networks has the same size as the input matrices except the number of channels. The output channels represent probabilities p_g and $1 - p_g$ since the softmax function is applied to the output.

Presented architectures (Fig. 6) differ in the type and number of layers used, dimension of convolutional kernels, and in the number of channels within each layer. Deconvolutional layers (previously also used in semantic segmentation [11]) were used in 3 of 4 presented topologies, including the best topology *L05+deconv*, which performs best in our experiments. In topologies *L05+deconv* and *L04-conv-dec*, the size of the convolutional layers is decreasing, when compared to the other two topologies. The effect of a significantly larger number of intermediate output channels is evaluated for topology *L03+deconv-inc-multich*.

The input of the CNN, which is prepared as described in Sec. III-A Eq. (3-7), is normalized and rescaled (8). This applies only to the depth d and the height p_y channels, since the intensity values of Velodyne sensor are already

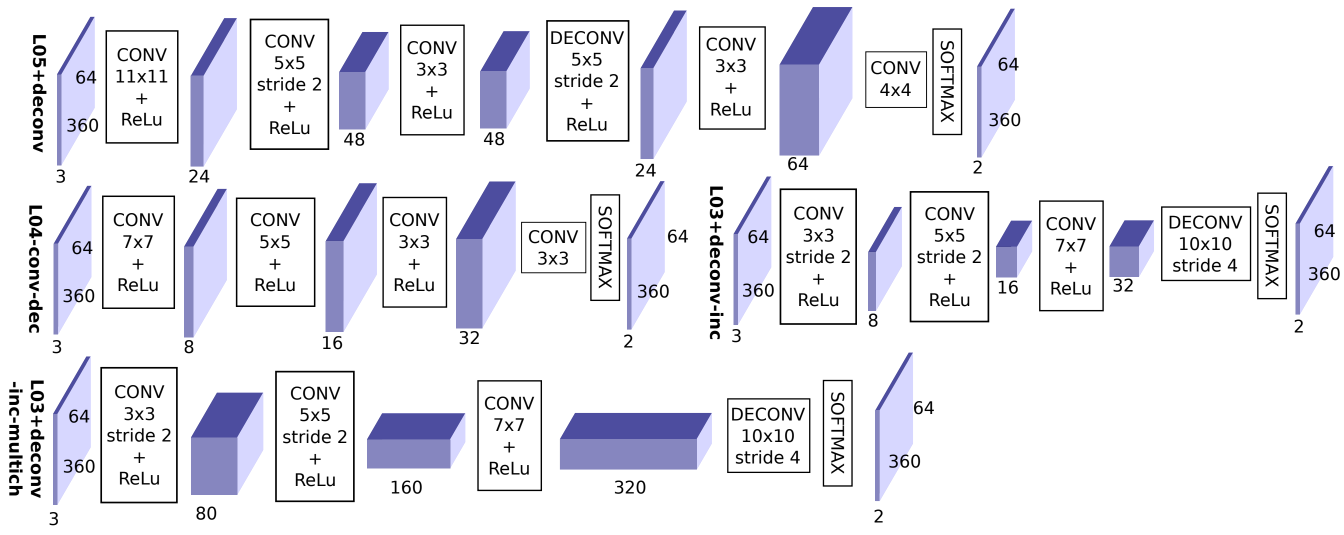

Fig. 6: Topology of the four proposed CNNs including dimensions of intermediate data blobs (blue blocks) and the number of channels below each blob. *L05+deconv* consists of 5 convolutional layers plus single deconvolution to restore the original frame width and height. *L04-conv-dec* process the input frame by 4 convolutional layers with decreasing size $(7, 5, 3, 3)$ of convolution kernel. In *L03+deconv-inc*, 3 convolutional layers with increasing kernel size are used. Deconvolution is used to restore original frame size in both this topology and in *L03+deconv-inc-multich* where the number of output channels are significantly larger comparing with other networks. Note: if the stride parameter N is set in (de-)colutional layers, the width and height of the output blob is (larger or) smaller N-times.

normalized to the interval $(0; 1)$. In our experiments, the normalization constant is set to $H = 3$, since in usual scenarios, the Velodyne model HDL-64E captures vertical slice approximately 3m high.

$$\overline{p_y} = \frac{p_y}{H}, \qquad \overline{d} = \log(d) \tag{8}$$

We applied this logarithmic rescaling for the depth channel to get approximately the same range differences between consequent rings for flat surfaces, both close and far from the sensor. The rescaling should suppress differences between the rings, due to varying distance from the sensor, and highlight those differences caused by the structure of observed scene – i.e. the obstacles (illustrated in Fig. 3). In the similar manner, the horizontal disparity was previously used as an input of a convolutional network, instead of using range value directly [12], what finally results in a normalization similar to ours.

IV. Experiments

The proposed convolutional networks were implemented, trained and evaluated using *Caffe*[4] deep learning framework. The human annotated dataset and the automatically annotated dataset were both used for training and pre-training of the proposed networks. We compared the results of our CNNs with the results of the robust state-of-the-art method [1] (using the original MATLAB implementation shared by the authors). It is necessary to mention one limitation of the Zhang's method. Because dimensions of the polar grid need to be set, the maximal range from the sensor is limited. In the experiments we used the 60m limit by default (and the

[4]http://caffe.berkeleyvision.org/

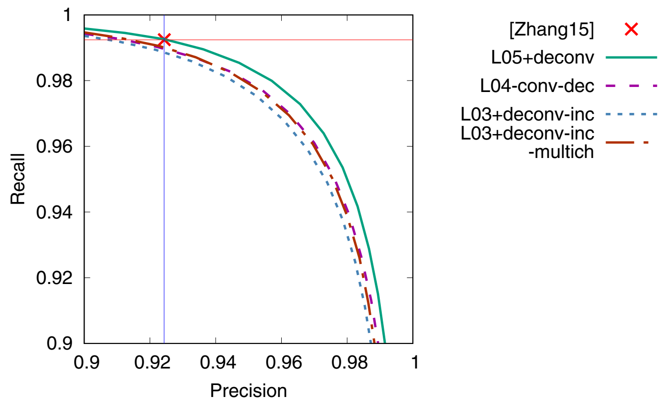

Fig. 7: The accuracy of the proposed networks and the reference method [1] for comparison. See Table I for numerical results.

30m limit in the time performance test). In order to make fair evaluation, we computed the accuracy of our method for both the maximal range set to 60m (same conditions as for [1]) and the unlimited range (to illustrate behavior for more distant measurements). Also, since the Zhang's method has no threshold/parameter for tuning the false positives to false negatives ratio, only a single precision/recall value can be computed instead of the whole PR curve.

Fig. 7 shows the comparison of different networks with the reference method [1]. The results are also summarized in Table I by means of the average precision and F-score as the metrics of accuracy. All networks were pre-trained using the automatically annotated data, trained and evaluated using the human annotated data and only the points within the range of 60m were taken into account.

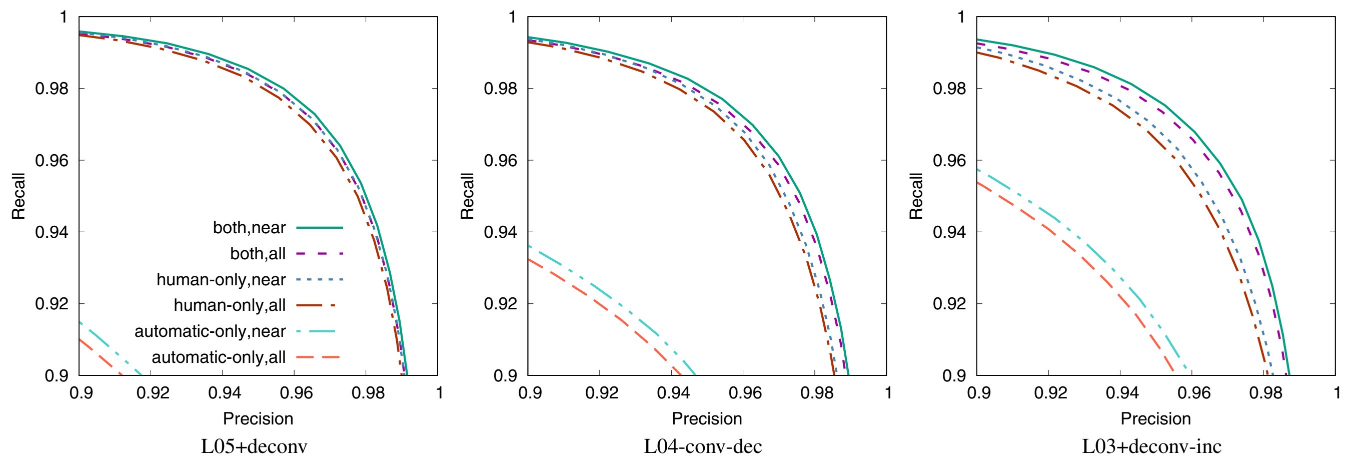

Fig. 8: Comparison of the different CNN topologies. The networks were trained using the human-made annotations (label *human-only*), artificial annotations (*automatic-only*) and both datasets for initialization and training (label *both*). *All* LiDAR points were processed, or only the points within the 60m range (label *near*) were taken into the account.

	AP	Precision* recall=.992	Recall* prec=.924	Best F-score
[Zhang15]	-	0.924	0.992	0.957
L05+deconv	**0.996**	**0.929**	**0.993**	**0.969**
L04-conv-dec	0.995	0.914	0.990	0.966
L03+deconv-inc	0.994	0.910	0.989	0.964
L03+deconv-inc-multich	0.995	0.916	0.990	0.966

TABLE I: Average precision (area under the PR curve), precision, recall and the best F-score of the proposed networks compared to [1]. *The precision (and the recall) was estimated for points where the recall (and precision respectively) is the same as the results of [1] (also displayed in Fig. 7 by red and blue line). The best F-score is taken as the highest value of harmonical average of precision and recall within the whole PR curve.

	CPU only [ms]	with GPU [ms]
L05+deconv	139	7.0
L04-conv-dec	90	3.2
L03+deconv-inc	8	1.2
L03+deconv-inc-multich	355	6.9

TABLE II: Performance comparison of the proposed networks in terms of speed. The average processing time per single Velodyne LiDAR HDL-64E frame is presented. The mini-batches of size 4 were used (i.e. 4 frames were processed in parallel).

The results (Fig. 7 and Table I) show that the accuracy is quite similar for different network topologies. Better accuracy is achieved with the networks where the size of convolution kernels decreases (*L05+deconv* and *L04-conv-dec* CNNs) and also with larger networks. The accuracy of *L05+deconv* network is also slightly higher compared to the reference method [1]. Preserving the same recall we were able to achieve 0.5% better precision and vice versa: 0.1% higher recall while preserving the same precision. Also, since our method enables balancing FP:FN ratio, we were able to find an optimal operating point yielding better F-score.

In Fig. 8, the precision-recall curves of different network topologies trained and evaluated in different ways are shown. We compared CNNs which were trained either by using the human-made annotations only (label *human-only*), or just by automatically annotated dataset (*automatic-only*), or by using both datasets together (label *both*). Moreover, we evaluated the accuracy of the situation in which all points are considered (label *all*), or when the maximal range is limited to 60m (label *near*) as used also by Zhang [1]. The examples of CNN outputs can be found in Fig. 9.

The results depicted in Fig. 8 show that cases in which reasoning about the ground was made only within the certain range (label *near*) yield better results. This is expected, since the density of measurements in farther areas is much lower. Also, the CNNs trained with human annotated datasets behave more accurately than CNNs trained on artificial data (evaluation is always made using the human annotations). An interesting fact is that this gap is less significant for networks with smaller architectures (e.g. *L03* compared to *L05*). This is probably caused by higher generalization which compromises discriminative power when learned on real annotations.

Table II shows the average processing time of proposed networks using CPU implementation (Intel i5-6500) and using GPU acceleration (GeForce GTX 770) on a standard desktop computer. These numbers indicate the usability of the networks for certain mobile robot platforms. *L03+decov-inc* requires low CPU consumption and therefore it is suitable also for small robots with low computational power. On the contrary, the *L05+deconv* topology would be suitable for platforms where GPU acceleration is available because of the superior accuracy.

As was said before, the main advantage of our method is superior time performance when compared to the method of

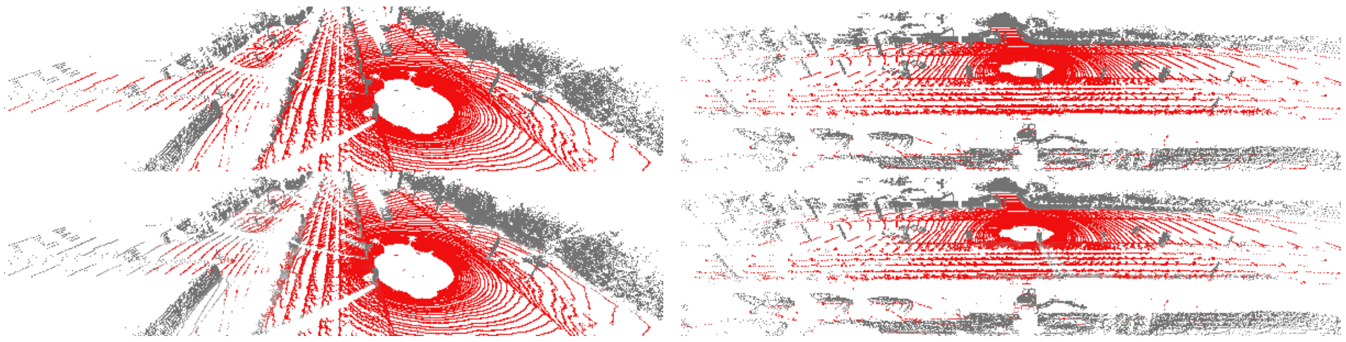
Fig. 9: Ground segmentations (outputs of CNN *L05+deconv*, bottom) for different LiDAR scans compared with human-made annotations (up). The results are near ideal but small differences are still visible under closer inspection.

Zhang et al. [1]. In our experiments, when using the Zhang's MATLAB implementation, the processing time of Velodyne HDL-64E LiDAR frame was 145 sec and consumed 11 GB of memory on average (note: no memory swapping which would compromise the performance happened during the experiments). Also, when we decreased the maximal range (and also the size of the internal 3D polar grid) to 30m, the processing time dropped to 75sec per frame and the memory consumption to approximately one half. However, this is still really far from real-time performance.

V. Conclusion

We presented a real time and robust ground segmentation method of Velodyne LiDAR data which outperforms the current state-of-the art methods in both the accuracy and speed. Our results show that the sparse LiDAR data can be encoded into its dense 2D representation and effectively processed by CNN. Our method improved the precision of state of the art [1] (by 0.5%) and significantly improved speed of the ground segmentation process from minutes to 140 ms using CPU and 7 ms with GPU acceleration.

In this paper we demonstrate that CNN approach is suitable for simpler task of ground segmentation where the results are near ideal. In the follow-up work, we want to explore the potential of this approach in more challenging semantic segmentation or move detection and also in quite different tasks of visual odometry estimation in the LiDAR data or point cloud registration.

As a secondary outcome of our work, we created the dataset with ground annotated and made it publicly available along with the annotation tool. Such data can be used for designing, training, and evaluation of other ground segmentation approaches.

References

[1] M. Zhang, D. D. Morris, and R. Fu, "Ground segmentation based on loopy belief propagation for sparse 3D point clouds," in *2015 International Conference on 3D Vision*, Oct 2015, pp. 615–622.

[2] D. Z. Wang, I. Posner, and P. Newman, "What could move? finding cars, pedestrians and bicyclists in 3D laser data," in *2012 IEEE Int. Conference on Robotics and Automation*, May 2012, pp. 4038–4044.

[3] C. Jiang, D. P. Paudel, Y. Fougerolle, D. Fofi, and C. Demonceaux, "Static-map and dynamic object reconstruction in outdoor scenes using 3-d motion segmentation," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 324–331, Jan 2016.

[4] G. Tanzmeister, J. Thomas, D. Wollherr, and M. Buss, "Grid-based mapping and tracking in dynamic environments using a uniform evidential environment representation," in *2014 IEEE Int. Conference on Robotics and Automation (ICRA)*, May 2014, pp. 6090–6095.

[5] Q. Li, L. Zhang, Q. Mao, Q. Zou, P. Zhang, S. Feng, and W. Ochieng, "Motion field estimation for a dynamic scene using a 3D LiDAR," *Sensors*, vol. 14, no. 9, pp. 16 672–16 691, 2014.

[6] J. Choi, S. Ulbrich, B. Lichte, and M. Maurer, "Multi-target tracking using a 3D-lidar sensor for autonomous vehicles," in *IEEE Int, Conference on Intelligent Transportation Systems*, Oct 2013, pp. 881–886.

[7] N. Wojke and M. Hselich, "Moving vehicle detection and tracking in unstructured environments," in *2012 IEEE International Conference on Robotics and Automation*, May 2012, pp. 3082–3087.

[8] A. Asvadi, P. Peixoto, and U. Nunes, "Detection and tracking of moving objects using 2.5d motion grids," in *IEEE Int. Conference on Intelligent Transportation Systems*, Sept 2015, pp. 788–793.

[9] C. Mertz, L. E. Navarro-Serment, MacLachlan, *et al.*, "Moving object detection with laser scanners," *J. Field Robot.*, vol. 30, no. 1, pp. 17–43, Jan. 2013. [Online]. Available: http://dx.doi.org/10.1002/rob.21430

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012.

[11] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *2015 IEEE CVPR*, 2015.

[12] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from rgb-d images for object detection and segmentation," in *European Conference on Computer Vision, Zurich, Switzerland*, 2014.

[13] B. Li, T. Zhang, and T. Xia, "Vehicle detection from 3D lidar using fully convolutional network," *CoRR*, vol. abs/1608.07916, 2016. [Online]. Available: http://arxiv.org/abs/1608.07916

[14] D. Kim, J. Sun, S. M. Oh, J. M. Rehg, and A. F. Bobick, "Traversability classification using unsupervised on-line visual learning for outdoor robot navigation," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, May 2006, pp. 518–525.

[15] A. Petrovskaya and S. Thrun, "Model based vehicle tracking for autonomous driving in urban environments," in *Robotics: Science and Systems IV, Eidgenössische Technische Hochschule Zürich, Zurich, Switzerland, June 25-28, 2008*, 2008.

[16] A. Petrovskaya and S. Thrun, "Efficient techniques for dynamic vehicle detection," in *Experimental Robotics, The Eleventh International Symposium, ISER 2008, July 13-16, Athens, Greece*, 2008, pp. 79–91.

[17] B. Douillard, A. Quadros, *et al.*, "Scan segments matching for pairwise 3D alignment," in *Robotics and Automation (ICRA), 2012 IEEE Int. Conference on*, May 2012, pp. 3033–3040.

[18] B. Douillard, J. Underwood, N. Kuntz, V. Vlaskine, A. Quadros, P. Morton, and A. Frenkel, "On the segmentation of 3D lidar point clouds," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 2798–2805.

[19] M. Velas, M. Spanel, and A. Herout, "Collar line segments for fast odometry estimation from velodyne point clouds," in *IEEE Int. Conference on Robotics and Automation*, May 2016, pp. 4486–4495.

[20] R. Zhang, S. A. Candra, K. Vetter, and A. Zakhor, "Sensor fusion for semantic segmentation of urban scenes," in *IEEE Int. Conference on Robotics and Automation (ICRA)*, May 2015, pp. 1850–1857.

Preprint submitted to 2018 IEEE International Conference on Robotics and Automation. Received September 14, 2017.