3D CNN-Based Semantic Labeling Approach for Mobile Laser Scanning Data

Balázs Nagy, *Student Member, IEEE*, and Csaba Benedek, *Senior Member, IEEE*

***Abstract*— In this paper, we introduce a 3D convolutional neural network (CNN)-based method to segment point clouds obtained by mobile laser scanning (MLS) sensors into nine different semantic classes, which can be used for high definition city map generation. The main purpose of semantic point labeling is to provide a detailed and reliable background map for self-driving vehicles (SDV), which indicates the roads and various landmark objects for navigation and decision support of SDVs. Our approach considers several practical aspects of raw MLS sensor data processing, including the presence of diverse urban objects, varying point density, and strong measurement noise of phantom effects caused by objects moving concurrently with the scanning platform. We also provide a new manually annotated MLS benchmark set called `SZTAKI CityMLS`, which is used to evaluate the proposed approach, and to compare our solution to various reference techniques proposed for semantic point cloud segmentation. Apart from point level validation we also present a case study on Lidar-based accurate self-localization of SDVs in the segmented MLS map.**

***Index Terms*— Semantic point cloud segmentation, deep learning, mobile laser scanning.**

I. Introduction

SELF-localization and scene understanding are key issues for self-driving vehicles (SDVs), especially in dense urban environments. Although the GPS-based position information is usually suitable for helping human drivers, its accuracy is not sufficient for navigating a SDV. Instead, the accurate position and orientation of the SDV should be calculated by registering the measurements of its onboard visual or range sensors to available 3D city maps [1].

Mobile laser scanning (MLS) platforms equipped with time synchronized Lidar sensors and navigation units can rapidly provide very dense and feature rich point clouds from large environments (see Fig. 1), where the 3D spatial measurements are accurately registered to a geo-referenced global coordinate system [2]–[4]. In the near future, these point clouds may act as a basis for detailed and up-to-date 3D High Definition (HD) maps of the cities, which can be be utilized by self driving vehicles for navigation, or by city authorities for road network management and surveillance, architecture or urban planning. However, all of these applications require semantic labeling of the data (Fig. 2). While the high speed of point cloud acquisition is a clear advantage of MLS, due to the huge data size yielded by each daily mission, applying efficient automated data filtering and interpretation algorithms in the processing side is crucially needed, which steps still introduce a number a key challenges.

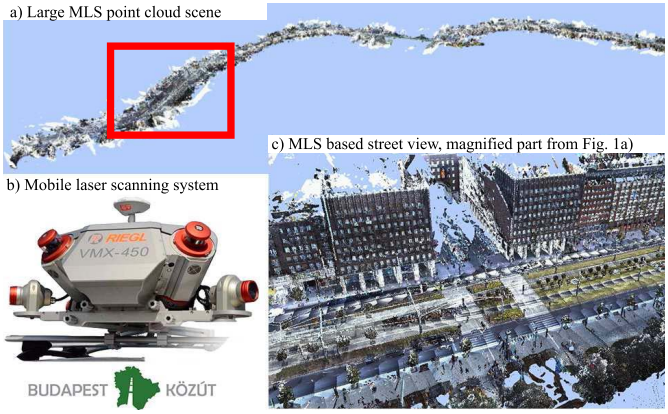


Fig. 1. MLS sensor and a scanned road segment.

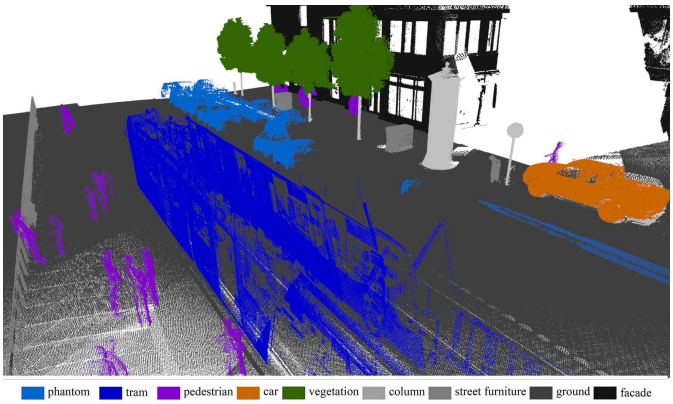


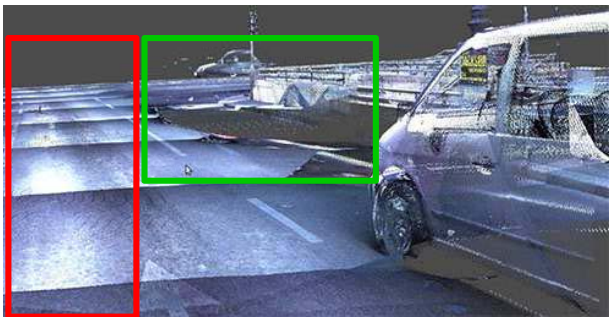
Fig. 2. Labeling result of the proposed 3D CNN based scene segmentation approach (test data provided by Budapest Közút Zrt.).

Manuscript received April 5, 2019; revised June 28, 2019; accepted June 30, 2019. Date of publication July 8, 2019; date of current version October 4, 2019. This work was supported in part by the National Research, Development and Innovation Fund under Grant NKFIA K-120233 and Grant KH-125681, in part by the Széchenyi 2020 Program under Grant EFOP-3.6.2-16-2017-00013 and Grant 3.6.3-VEKOP-16-2017-00002, and in part by the National Excellence Program under Grant 2018-1.2.1-NKP-00008. The work of C. Benedek was supported by the Janos Bolyai Research Scholarship of the Hungarian Academy of Sciences. The associate editor coordinating the review of this paper and approving it for publication was Dr. Edward Sazonov. *(Corresponding author: Balázs Nagy.)*

The authors are with the Machine Perception Research Laboratory, MTA SZTAKI, 1111 Budapest, Hungary, and also with Pázmány Péter Catholic University, 1083 Budapest, Hungary (e-mail: nagy.balazs@sztaki.mta.hu; benedek.csaba@sztaki.mta.hu).

Digital Object Identifier 10.1109/JSEN.2019.2927269

(a) Raw MLS data with phantoms



(b) Result of phantom removal

Fig. 3. Demonstration of the phantom effect in MLS data and the result of phantom removal with the proposed approach.

Taking the raw MLS measurements, one of the critical issues is the *phantom* effect caused by independent object motions (Fig. 3). Due to the sequential nature of the environment scanning process, scene objects moving concurrently with the MLS platform (such as passing vehicles and walking pedestrians) appear as phantom-like longdrawn, distorted structures in the resulting point clouds [5]. It is also necessary to recognize and mark all movable scene elements such as pedestrians, parking vehicles [3] or trams from the MLS scene. On one hand, they are not part of the reference background model, thus these regions must be eliminated from the HD maps. On the other hand, the presence of these objects may indicate locations of sidewalks, parking places etc. Column-shaped objects, such as poles, traffic sign bars [2], tree trunks are usually good landmark points for navigation. Finally, vegetation areas (bushes, tree foliage) should also be specifically labeled [4]: since they are dynamically changing over the whole year, object level change detection algorithms should not take they into account.

To address the above complex multi-class semantic labeling problem we introduce a new 3D convolutional neural network (CNN) based approach to segment the scene in voxel level, and for testing the approach, we present the `SZTAKI CityMLS` benchmark set containing different labeled scenes from dense urban environment. Differently from previously proposed general point cloud labeling frameworks [6], [7], the present approach is focusing on challenging issues of MLS data processing in self-driving applications. For this reason, apart from a detailed comparative evaluation of the proposed segmentation method versus existing reference techniques, we also present a case study on Lidar based accurate self-localization of SDVs in the segmented MLS map, showing qualitatively and quantitatively the advantages of the improvements.

II. Related Work

While a number of various approaches have already been proposed for general point cloud scene classification, they are not focusing on all practical challenges of the above introduced worklfow of 3D map generation from raw MLS data. In particularly, only a few related works have discussed the problem of *phantom* removing. Point-level and statistical feature based methods such as [8] and [9] examine the local density of a point neighborhood, but as noted in [10] they do not take into account higher level structural information, limiting the detection rate of *phantoms*. The task is significantly facilitated if the scanning position (e.g. by tripod based scanning [11]) or a relative time stamp (e.g. using a rotating multi-beam Lidar [12]) can be assigned to the individual points or point cloud frames, which enables the exploitation of multi-temporal feature comparison. However, in case of our examined MLS point clouds no such information is available, and all points are represented in the same global coordinate system.

Several techniques extract various object blob candidates by geometric scene segmentation [2], [13], then the blobs are classified using shape descriptors, or deep neural networks [13]. Although this process can be notably fast, the main bottleneck of the approach is that it largely depends on the quality of the object detection step.

Alternative methods implement a voxel level segmentation of the scene, where a regular 3D voxel grid is fit to the point cloud, and the voxels are classified into various semantic categories such as roads, vehicles, pole-like objects, etc. [4], [14], [15]. Here a critical issue is feature selection for classification, which has a wide bibliography. Handcrafted features are efficiently applied by a maximum-margin learning approach for indoor object recognition in [16]. Covariance, point density and structural appearance information is adopted in [17] by a random forest classifier to segment MLS data with varying density. However, as the number and complexity of the recognizable classes increase, finding the best feature set by hand induces challenges.

3D CNN based techniques have been widely used for point cloud scene classification in the recent years, following either *global* or *local* (window based) approaches. *Global* approaches consider information from the complete 3D scene for classification of the individual voxels, thus the main challenge is to keep the time and memory requirements tractable in large scenes. The OctNet method implements a new complex data stucture for efficient 3D scene representation which enables the utilization of deep and high resolution 3D convolutional networks [6]. From a practical point of view, by OctNet's training data annotation operators should fully label complete point cloud scenes, which might be an expensive process.

Sliding window based techniques are usually computationally cheaper, as they move a 3D box over the scene, using locally available information for the classification of each

point cloud segment. The *Vote3Deep* [14] assumes a fixed-size object bounding box for each class to be recognized, which might be less efficient if the possible size range of certain objects is wide. A CNN based voxel classification method has recently been proposed in [15], which uses purely local features, coded in a 3D occupancy grid as the input of the network. Nevertheless, they did not demonstrate the performance in the presence of strong *phantom* effects, which require accurate local density modeling [9], [10].

The multi-view technique [18] projects the point cloud from several (twelve) different viewpoints to 2D planes, and trains 2D CNN models for the classification. Finally, the obtained labels are backprojected to the 3D point cloud. This approach presents high quality results on synthetic datasets and in point clouds from factory environments, where due to careful scanning complete 3D point cloud models of the scene objects are available. Application for MLS data containing partially scanned objects is also possible, but the advantages over competing approaches are reduced here [18].

PointNet++ [7] introduces a hierarchical neural network for point set classification. The method takes random samples within a given radius of the examined point, so it does not exploits density features. Results are demonstrated on synthetic and indoor data samples, with dense and accurate spatial data and RGB color information.

The *Similarity Group Proposal Network* (SGPN) [19] uses PointNet++ as a backbone feature extractor, and presents performance improvement by adding several extra layers to the top of the network structure. However as noted by the authors, SGPN cannot process large scenes on the order 10^5 or more points [19], due to using a similarity matrix whose size scales quadratically as the number of points increases. This property is disadvantageous for MLS data processing, where a typical scene may contain over 10^7 points.

The *Sparse Lattice Network* (SPLATNet$_{3D}$) [20] is a recent technique which able to deal with large point cloud scenes efficiently by using a Bilateral Convolution Layer (BCL). SPLATNet$_{3D}$ [20] projects the extracted features to a lattice structure, and it applies sparse convolution operations. Similarly to voxel based approaches, the lattice structure implements a discrete scene representation, where one should address under- and oversegmentation problems depending on the lattice scales.

III. Benchmark Issues

A number of benchmark sets have already been published for 3D point cloud segmentation in urban environment, including MLS datasets Oakland [21] (1.6M points), Paris-rue-Madame (20M points) [22] and data from the IQmulus & TerraMobilita Contest (12M labeled points) [23]. However, their available annotated segments are relatively small, which make the development of supervised classification algorithms less relevant due to overfitting problems.

The Semantic3D.net benchmark [24] contains a considerable larger set of labeled data, however it has been created with static terrestrial laser scanners (TLS) which produce more accurate and in certain regions significantly denser point clouds than MLS. As shown in Fig. 4 the point density of

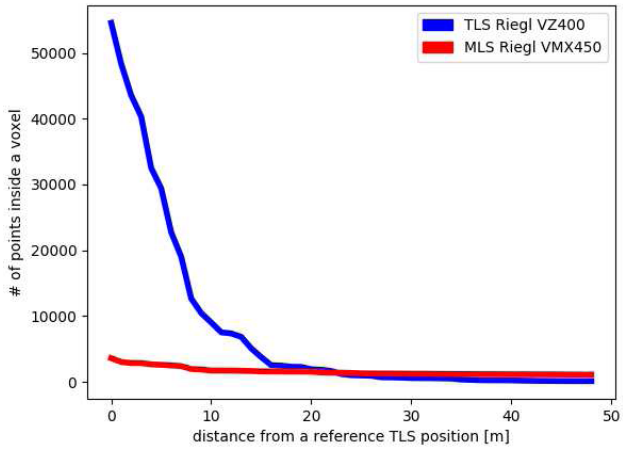


Fig. 4. Point cloud characteristics comparison of measurements from the same region obtained by a static Riegl VZ-400 TLS sensor and a moving Riegl VMX-450 MLS system, respectively. Point density is displayed as a function of the distance from the TLS sensor's center position.

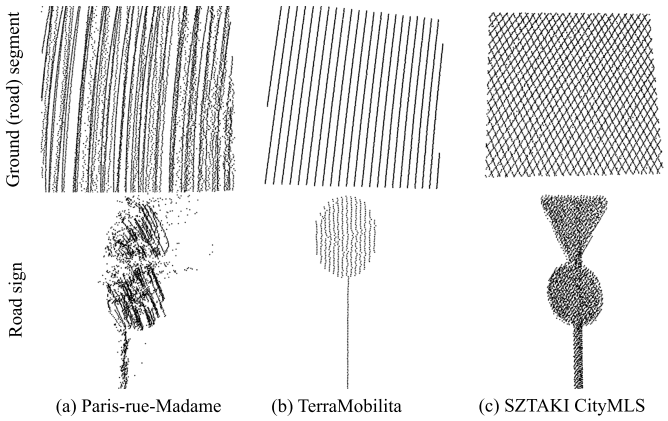
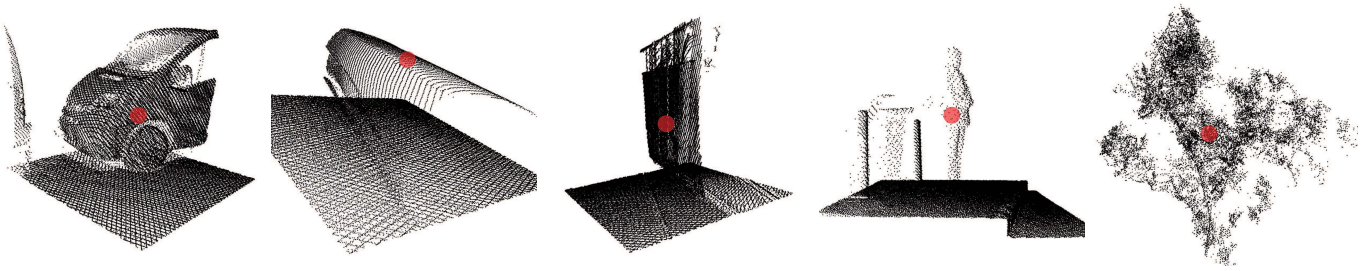


Fig. 5. Data quality comparison between two reference datasets and the proposed `SZTAKI CityMLS` dataset.

a single TLS sensor is steeply decreasing as a function of the distance from the sensor, while applying mobile scanning, we can obtain a more uniform, but generally lower point density in the same region. In addition, the density characteristic of a large point cloud segment obtained by TLS from multiple scanning positions is strongly varying, since TLS operators may follow arbitrary trajectories and timing constraints during the scanning mission. Therefore, comparing two different TLS datasets may show significant differences, even if they have been recorded by the same scanner, but for different purpose or by different operators. As a consequence, developing widely usable object detection methods for large-scale TLS datasets needs careful practical considerations.

On the other hand, MLS scene segmentation is today a highly relevant field of research, with strong industrial interest. In MLS data recording, the car passes with a normal 30-50 km traveling speed, following a more predictable trajectory (usually scans are preformed in both directions for a two-way road), therefore the effects of the driving dynamics on the obtained point cloud can be indirectly incorporated into the learning process. However, compared to TLS data, ghost filtering is more difficult, and the measurement noise is higher.



(a) Static car (b) Phantom (moving car) (c) Tram (d) Pedestrian (e) Vegetation

Fig. 6. Different training volumes extracted from point cloud data. Each training sample consists of $K \times K \times K$ voxels (used $K = 23$), and they are labeled according to their central voxel (highlighted with red).

In this paper, we utilize MLS data captured by a Riegl VMX-450 for real industrial usage by the Road Management Department of the Budapest City Council. Our new SZTAKI CityMLS dataset contains in total around 327 Million annotated points from various urban scenes, including main roads with both heavy and solid traffic, public squares, parks, and sidewalk regions, various types of cars, trams and buses, several pedestrians and diverse vegetation.

As shown in Fig. 5 the data characteristic of SZTAKI CityMLS is significantly different from TerraMobilita and Paris-rue-Madam data, making the proposal of the new benchmark indeed relevant. While Paris-rue-Madame database contains the most dense point clouds, due to registration issues of the recorded Rotating Multi-beam Lidar (Velodyne) frames, the obtained point cloud is quite noisy. On the other hand, the TerraMobilita database was captured with multiple 2D laser scanners yielding accurate spatial point cloud coordinates, but the measurements are sparse: depending on the speed of the scanning platform smaller objects may be composed of a few line segments only. As for SZTAKI CityMLS, the Riegl VMX-450 scans are well suited to industrial applications requiring dense, accurate and notable homogeneous point clouds.

IV. Proposed Approach

In the paper, we propose a new 3D CNN based semantic point cloud segmentation approach, which is adopted to dense MLS point clouds of large scale urban environments, assuming the presence of high variety of objects, with strong and diverse *phantom* effects. The present technique is based on our earlier model [5] specifically developed for phantom detection and removal, which we extend here for recognizing nine different semantic classes required for 3D map generation: *phantom*, *tram/bus*, *pedestrian*, *car*, *vegetation*, *column*, *street furniture*, *ground* and *facade*. As main methodological differences from [5], our present network uses a two channel data input derived from the raw MLS point cloud featuring local point density and elevation; and a voxel based space representation, which can handle the separation of tree crowns or other hanging structures from ground objects more efficiently than the pillar based model of [5]. To keep the computational requirements low, we implemented a sparse voxel structure avoiding unnecessary operations on empty space segments.

A. Data Model for Training and Recognition

Data processing starts with building our sparse voxel structure for the input point cloud, with a fine resolution (used $\lambda = 0.1m$ voxel side length). During classification we will assign to each voxel a unique class label from our nine-element label set, based on majority votes of the points within the voxel.

Next we assign two feature channels to the voxels based on the input cloud: point *density*, taken as the number of included points, and *mean elevation*, calculated as the average of the point height values in the voxel.

The unit of training and recognition in our network is a $K \times K \times K$ voxel neighborhood (used $K = 23$), called hereafter training volume. To classify each voxel v, we consider the point density and elevation features in all voxels in the v-centered training volume, thus a given voxel is labeled based on a 2-channel 3D array derived from K^3 local voxels. The proposed 3D CNN model classifies the different training volumes independently. This fact specifies the roles of the two feature channels: while the *density* feature contributes to model the local point distribution within each semantic class, the *elevation channel* informs us about the expected (vertical) locations of the samples regarding the different categories, providing impression from the global position of the data segment within the large 3D scene.

Fig. 6 demonstrates various training volumes, used for labeling the central voxel highlighted with red color. As we consider relatively large voxel neighborhoods with $K \cdot \lambda$ (here: $2.3m$) side length, the training volumes often contain different segments of various types of objects: for example, Fig. 6(b) contains both phantom and ground regions, while Fig. 6(d) contains column, ground and pedestrian regions. These variations add supplementary contextual information to the training phase beyond the available the density and elevation channels, making the trained models stronger.

B. 3D CNN Architecture and Its Utilization

Our proposed 3D CNN network implements an end-to-end pipeline: the feature extractor part (combination of several 3D convolution, max-pooling and dropout layers) optimizes the feature selection, while the second part (fully connected dense layers) learns the different class models. Since the size of the training data ($23 \times 23 \times 23$) and the number of classes (9)

TABLE I

PERFORMANCE ANALYSIS OF THE PROPOSED C^{2}CNN METHOD AS A FUNCTION OF THE VOXEL SIZE PARAMETER

Parameters	Voxel size λ [m], using a fixed $K = 23$ kernel size						
	0.02	0.05	0.1	0.2	0.3	0.4	0.5
Number of voxels	812500000	52000000	6500000	812500	240596	101563	52000
Precision	34.7	77.8	90.4	83.6	76.3	64.2	44.7
Recall	29.8	69.7	90.2	85.9	77.8	61.7	48.5
F measure	32.1	73.5	90.3	84.7	77.0	62.9	46.5

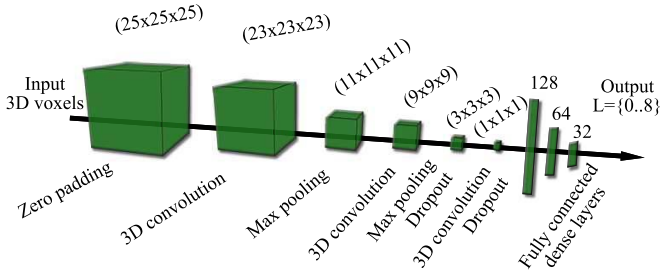


Fig. 7. Structure of the proposed 3D convolutional neural network, containing three 3D convolutional layers, two max-pooling and two dropout layers. The input of the network is a $K \times K \times K$ voxel (used $K = 23$) data cube with two channels, featuring density and point altitude information. The output of the network is an integer value from the set $L = 0..8$.

are quite small, we construct a network with a similar structure to the well known LeNet-5 [25], with adding an extra convolution layer and two new dropout layers to the LeNet-5 structure, and exchanging the 2D processing units to the corresponding 3D layers. Fig. 7 demonstrates the architecture and the parameters of the trained network. Each convolution layer uses $3 \times 3 \times 3$ convolution kernels and a Rectified Linear Unit (ReLu) activation function, while the numbers of filters are 8, 16 and 32 in the 1st, 2nd and 3rd convolution layer, respectively. The output layer is activated with a Softmax function. To avoid overfitting, we use dropout regularization technique, randomly removing 30% of the connections in the network. Moreover to make our trained object concepts more general, we clone and randomly rotate the training samples around their vertical axis several times. The network is trained with Stochastic Gradient Descent (SGD) algorithm, and we change the learning rate in the training epochs as a function of the validation accuracy change.

To segment a scene, we move a sliding volume across the voxelized input point cloud, and capture the $K \times K \times K$ neighborhood around each voxel. Each neighborhood volume is separately taken as input by the CNN classifier, which predicts a label for the central voxel only. As the voxel volumes around the neighboring voxels strongly overlap, the resulting 3D label map is usually smooth, making possible object or object group extraction with conventional region growing algorithms (see Fig. 2, 8).

V. EXPERIMENTAL RESULTS AND EVALUATION

A. Point Cloud Annotation and Training

Large-scale MLS scene annotation is a crucial step in deep learning based approaches. For this reason, we developed a user friendly 3D point cloud annotator tool, that allows operators to label arbitrary shaped 3D volumes quickly. We assigned unique labels to *occupied* voxels of the scene, using 10cm voxels which determines the spatial accuracy of annotation. In one step, the operator can mark a rectangle area on the screen, which defines with the actual viewpoint a 3D pyramid volume in scene's 3D coordinate system. Then, the annotated volume can be created through a combination of union and intersection operations on several pyramids. With this tool we manually labeled around 327M points over a 30.000 m^2 area of the city, with more than 50m elevation differences, using the earlier defined nine classes. As a result of annotation, we created a new benchmark set called `SZTAKI CityMLS`.[1]

Next, we divided our data into three non-overlapping segments used for training, validation and test, respectively. For training data generation, we randomly selected 100.000 voxels from each class's representative region in the *training segment* of the labeled data, and extracted the 2-channel $K \times K \times K$ voxel volumes around each training sample, which were used as the local fingerprints of the corresponding point cloud parts. This selection yielded in total 900.000 volumes, used for training the network. During the training process, we tuned the parameters of the classifier on a validation set, which contains 20.000 samples from each class, selected from the *validation segment* of the point cloud.

The quantitative performance evaluation of the network is performed on an independent test set (without any overlap with the training and the validation sets), including two million voxel volumes extracted from the *test segment* of the point cloud, representing the classes evenly.

B. Hyperparameter Tuning

Voxel size λ and dimension of the data sample cube (K) are two important hyperparameters of the proposed model, which have to be carefully tuned with respect to the data density and the recognizable classes. We have optimized these parameter with a *grid search* algorithm, which yielded an optimum of $\lambda = 0.1$m and $K = 23$, regarding our Riegl VMX-450 data, as mentioned in Sec. IV-A. For further analysis, Table I shows the model performance as a function of different λ voxel size settings, with a fixed $K = 23$ value. We can observe a maximal performance at $\lambda = 0.1$m. Using smaller voxels, the model tends to oversegment the scene, while adopting a too large voxel size, the CNN-based label prediction yields coarse region boundaries.

[1]The `SZTAKI CityMLS` dataset is available at the following url: http://mplab.sztaki.hu/geocomp/SZTAKI-CityMLS-DB.html

TABLE II

PERFORMANCE ANALYSIS OF THE PROPOSED C^{2}CNN METHOD AS A FUNCTION OF THE DATA CUBE SIZE $K \times K \times K$

Parameters	Data cube's side length (K), using a fixed 0.1 m voxel size										
	7	11	17	21	23	25	27	29	31	37	41
Precision	58.4	72.5	81.1	87.6	90.4	88.5	86.4	83.2	78.9	72.8	69.4
Recall	55.7	69.7	82.6	87.1	90.2	89.1	87.2	85.6	82.2	71.2	69.6
F measure	57.0	71.1	81.8	87.4	90.3	88.8	86.8	84.4	80.5	72.0	69.5

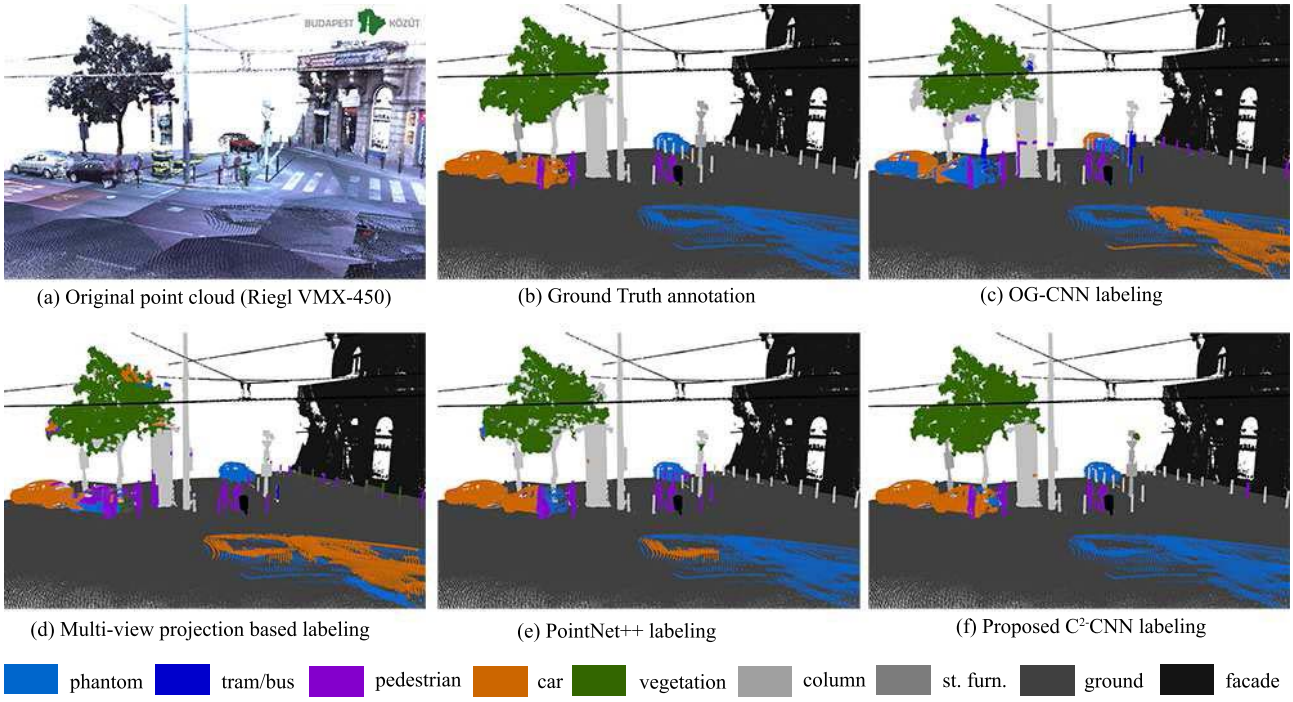


Fig. 8. Qualitative comparison of the results provided by the three reference methods, (c) OG-CNN, (d) Multi-view approach and (e) PointNet++, and the proposed (f) C^{2}CNN approach in a sample scenario. For validation, Ground Truth labeling is also displayed in (b).

On the other hand, Table II demonstrates the dependence of the results on the data cube's side length (K), with choosing a constant voxel grid resolution of $\lambda = 0.1$m. Using significantly smaller kernels than the optimal $K = 23$, the model can only consider small local voxel neighborhoods, which do not enable efficient contextual modeling. However, in cases of too large kernels, the training/test samples may contain significant noise and irrelevant background segments, which fact often leads to overfitting problems.

C. Evaluation and Comparison to Reference Techniques

We evaluated our proposed method against four reference techniques in qualitative and quantitative ways on the `SZTAKI CityMLS` dataset. First, we tested a single channel 3D CNN [15], which uses a 3D voxel occupancy grid (OG) as input (OG-CNN). Second, we implemented a multi-view method based on [18], that projects the point cloud onto different planes, and achieves CNN classification in 2D. Third, we tested the PointNet++ [7] deep learning framework, using their publicly available source code. Finally we adopted the implementation of SPLATNet$_{3D}$ [20], by applying two different feature selection strategies.

Fig. 8 shows a sample scene for qualitative comparison of the manually edited Ground Truth, the outputs of the OG-CNN, multi-view and PointNet++ methods, and the result of the proposed two channel C^{2}CNN technique. We also evaluated the proposed and the reference methods in a quantitative way. Table III shows the voxel level precision (Pr), recall (Rc) and F-rates (F-r) for each class separately as well as the overall performance weighted with the occurrence of the different classes. Note that Table III does not contain the values obtained regarding *facades* and *ground*, which classes proved to be quite easy to recognize for the CNN network (over 98% rates), thus their consideration could yield overrating the performance of the object discrimination abilities of the method.

By analyzing the results, we can conclude that the proposed C^{2}CNN can classify all classes of interest with an F-rate larger than 83%. The precision and recall rates for all classes are quite similar, thus the false negative and false positive hits are nearly evenly balanced. The two most efficiently detected classes are the tram/bus, whose large planar sides are notably characteristic, and vegetation, which usually correspond to unorganized point cloud segments on predictable positions (bushes on street level and tree crowns at higher altitude).

TABLE III

QUANTITATIVE EVALUATION OF THE PROPOSED C^{2}CNN APPROACH AND THE REFERENCE TECHNIQUES ON THE NEW SZTAKI CityMLS DATASET

Class	OG-CNN [15]			Multi-view [18]			PointNet++ [7]			SPLATNet^{xyz} [20]			SPLATNet$^{xyz}_{rgb}$ [20]			Proposed C^{2}CNN		
	Pr	Rc	F-r	Pr	Rc	F-r	Pr	Rc	F-r	Pr	Rc	F-r	Pr	Rc	F-r	Pr	Rc	F-r
Phantom	85.3	34.7	49.3	76.5	45.3	56.9	82.3	76.5	79.3	82.5	80.9	81.7	83.4	78.2	80.7	84.3	85.9	**85.1**
Pedestrian	61.2	82.4	70.2	57.2	66.8	61.6	86.1	81.2	83.6	82.6	82.1	82.3	80.4	78.6	79.5	85.2	85.3	**85.2**
Car	56.4	89.5	69.2	60.2	73.3	66.1	80.6	92.7	86.2	81.5	90.0	85.5	81.1	89.4	85.0	86.4	88.7	**87.5**
Vegetation	72.4	83.4	77.5	71.7	78.4	74.9	91.4	89.7	90.5	87.1	88.2	87.6	86.4	87.3	86.8	98.2	95.5	**96.8**
Column	88.6	74.3	80.8	83.4	76.8	80.0	83.4	93.6	**88.2**	84.3	90.2	87.2	84.1	89.2	86.6	86.5	89.2	87.8
Tram/Bus	91.4	81.6	86.2	85.7	83.2	84.4	83.1	89.7	86.3	82.1	83.5	82.8	79.3	82.1	80.7	89.5	96.9	**93.0**
Furniture	72.1	82.4	76.9	57.2	89.3	69.7	84.8	82.9	83.8	84.7	86.2	**85.4**	82.6	81.3	81.9	88.8	78.8	83.5
Overall	76.9	74.2	75.5	72.5	73.4	72.9	85.6	87.5	86.5	83.5	85.9	84.7	82.5	83.7	83.0	90.4	90.2	**90.3**

Note: Voxel level Precision (Pr), Recall (Rc) and F-rates (F-r) are given in percent (overall values weighted with class significance)

TABLE IV

QUANTITATIVE COMPARISON OF THE PROPOSED METHOD AND THE REFERENCE ONES ON THE *Oakland* DATASET. F-RATE VALUES ARE PROVIDED IN PERCENT

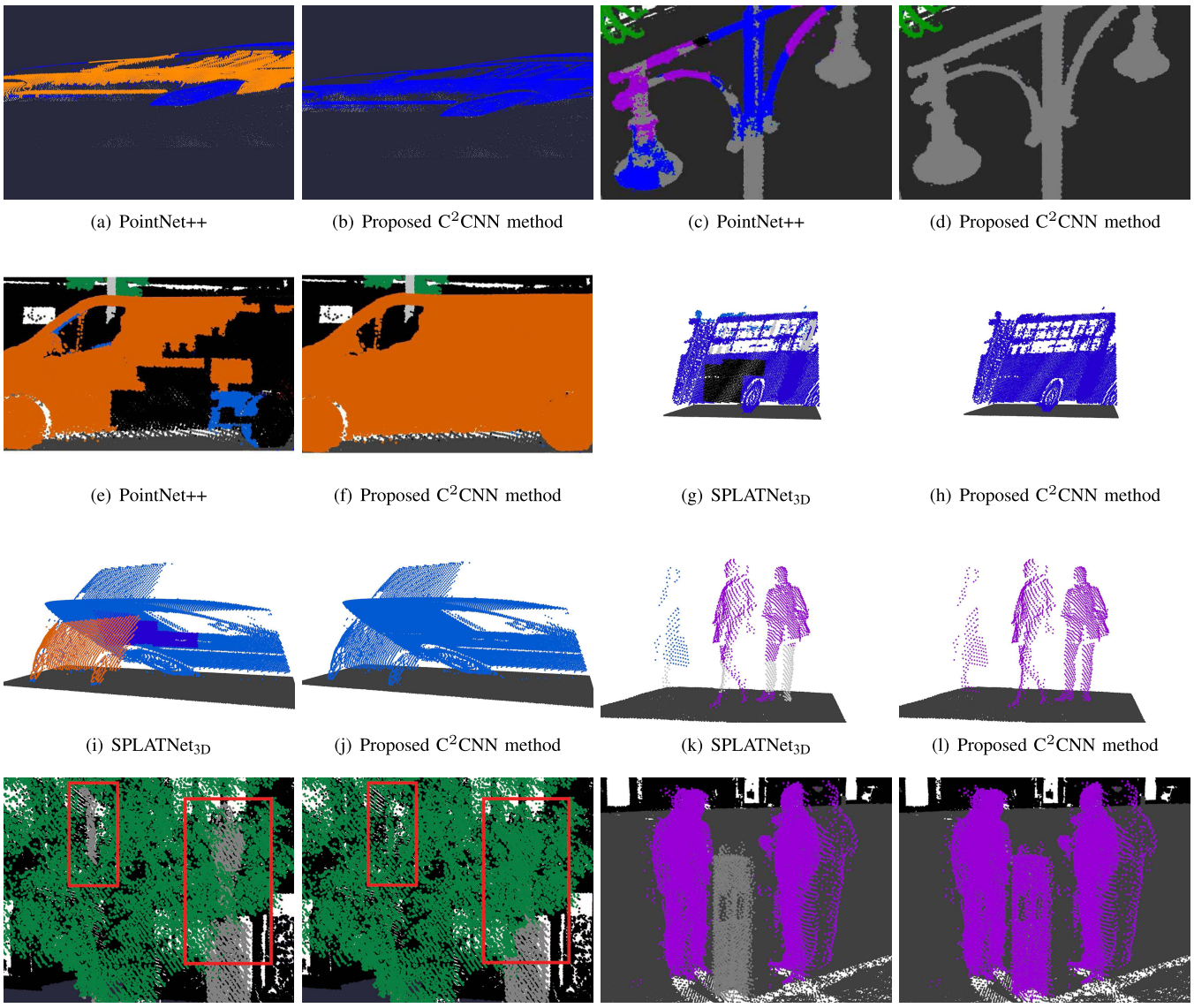
Class	Markov [21]	PointNet++ [7]	OG-CNN [15]	Multi-view [18]	SPLATNet [20]	Proposed C^{2}CNN
Vegetation	97.2	91.1	87.3	70.4	84.2	96.5
Ground	96.1	91.8	88.8.1	73.4	92.9	98.6
Facade	95.7	96.3	80.7	68.7	90.1	97.7
Pole-like	64.3	79.2	52.1	45.9	70.6	73.3
Vehicle	67.8	68.0	59.4	60.5	66.2	74.7
Street fur.	59.3	73.4	64.7	59.2	66.8	71.4

Nevertheless, classes with high varieties such as phantoms, pedestrians and cars are detected with 85-87% F-rates, indicating balanced performance over the whole scene.

Since SPLATNet is able to consider both geometry and color information associated to the points, we tested this approach with two different configurations. SPLATNet^{xyz} deals purely with the Euclidean point coordinates (similarly to C^{2}CNN and all other listed reference techniques), while SPLATNet$^{xyz}_{rgb}$ also exploits *rgb* color values associated to the points. As the results confirm in the considered MLS data SPLATNet^{xyz} proved to be slightly more efficient, which is a consequence of the fact, that automated point cloud texturing is still a critical issue in industrial mobile mapping systems, which is affected by a number of artifacts. The overall results of the four reference techniques fall behind our proposed method with a margin of 14.8% (OG-CNN), 17.4% (multi-view), 3.8% (PointNet++), and 5% (SPLATNet^{xyz}) respectively. While the overall Pr and Rc values of all references are almost equal again, there are significant differences between the recognition rates of the individual classes. The weakest point of all competing methods is the recall rate of phantoms, which class has diverse appearance in the real measurements due to the varying speed of both the street objects and the scanning platform. For (static) cars, the recall rates are quite high everywhere, but due to their confusion with phantoms, there are also many false positive hits yielding lower precision. By OG-CNN, many pedestrians are erroneously detected in higher scene regions due to ignoring the elevation channel, which provides some global position information for the C^{2}CNN model, meanwhile preserving the quickness of detection through performing local calculations only.

Apart from the above detailed evaluation on the SZTAKI CityMLS dataset, we also tested our method on various existing point cloud benchmarks mentioned in Sec. III. On one hand, we trained the C^{2}CNN method on the annotated part of the *TerraMobilita* dataset [23], and predicted the class labels for different test regions. Some qualitative results of classification are shown in Fig. 10, which confirm that our approach could be suited to this sort of sparser measurement set as well, however the number of annotated street objects for training should be increased to enhance the results. We can expect similar issues regarding the *Paris-rue-Madame* dataset [22], while our model does not suite well the *Semantic3D.net* data [24], where the point cloud density is drastically varying due to usage of static scanners.

Next, we demonstrate that our method can also be adopted to the *Oakland* point clouds [21]. Since that dataset is very small (1.6M points overall), we took a C^{2}CNN network pre-trained on our SZTAKI CityMLS dataset, and fine tuned the weights of the model using the training part of the *Oakland* data. Generally, the *Oakland* point clouds are sparser, but have a more homogeneous density than SZTAKI CityMLS. As sample results in Fig. 11 confirm, our proposed approach can efficiently separate the different object regions here, although some low-density boundary components of the vehicles may erroneously identified as phantoms. Using the *Oakland* dataset, we can also provide quantitative comparison between the C^{2}CNN method, the reference techniques from Table III, and also the Max-Margin Markov Network (Markov) based approach presented in [21]. Table IV shows again the superiority of C^{2}CNN over all references. Both Markov [21] and the C^{2}CNN methods are able to identify the vegetation, ground and facade regions with around 95-98% accuracy, but for pole-like objects, street furniture and vehicles the proposed method outperforms the reference technique with 8-10%.



(a) PointNet++ (b) Proposed C^{2}CNN method (c) PointNet++ (d) Proposed C^{2}CNN method

(e) PointNet++ (f) Proposed C^{2}CNN method (g) SPLATNet$_{3D}$ (h) Proposed C^{2}CNN method

(i) SPLATNet$_{3D}$ (j) Proposed C^{2}CNN method (k) SPLATNet$_{3D}$ (l) Proposed C^{2}CNN method

(m) Ground truth (n) Proposed C^{2}CNN method (o) Ground truth (p) Proposed C^{2}CNN method

Fig. 9. Typical failure cases of the proposed and the reference methods.

D. Failure Case Analysis of the PointNet++ the SPLATNet$_{3D}$ and the Proposed C^{2}CNN Methods

In Fig. 9, we demonstrate typical failure scenarios of PointNet++, SPLATNet$_{3D}$ and the proposed C^{2}CNN, which are the three most successful methods according to Table III. Experimental performance evaluation of PointNet++ and SPLATNet$_{3D}$ in their presenting articles [7], [20] has been restricted to indoor scenes, synthetic databases, or TLS based facade point clouds which are not affected by motion artifacts or heavy occlusion effects. As emphasized in Sec. I and III MLS data of the new `SZTAKI CityMLS` benchmark has significantly different characteristic from the existing datasets, and it presents particularly challenging issues such as phantoms, incomplete object segments and multiple occlusions between street objects and the 3D background scene.

Some limitations of the PointNet++ approach are shown in three point could segments in Fig. 9(a)-(f) with comparative results obtained by proposed C^{2}CNN technique. Since PointNet++ is trained on local point neigborhoods, large phantom regions with inhomogeneous point density often mislead the process (Fig. Fig. 9(a)). On the other hand, without explicitly considering the global position information, pedestrians, phantoms or street furniture may be also detected on the height level of the tree crowns or street lamps (Fig. 9(c)). In other cases large planar vehicle parts may be confused with building facades (Fig. 9(e)).

By testing the SPLATNet$_{3D}$, we have observed that even with optimized lattice scale settings, it often oversegments compact objects such as vehicles (Fig. 9(g), 9(i)). For example it may classify the border of a bus window as a column, or classify some planar vehicles segments as facade. In addition, a number of pedestrians are divided into two different regions (Fig. 9(k)), so that their upper parts are correctly predicted as pedestrians, but the legs are recognized as columns.

Minor over- or undersegmentation issues may also appear by applying the proposed C^{2}CNN approach, however we have

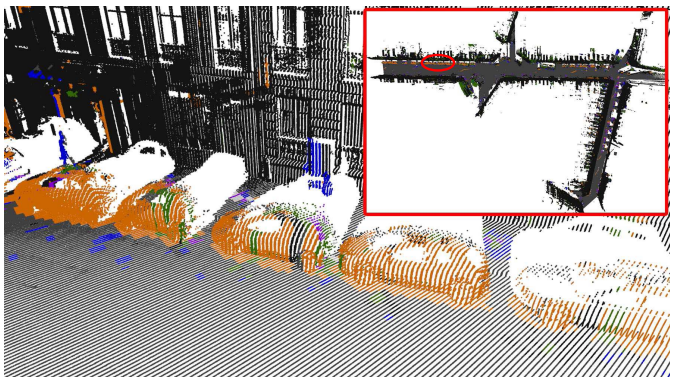


Fig. 10. Test result on the TerraMobilita data.

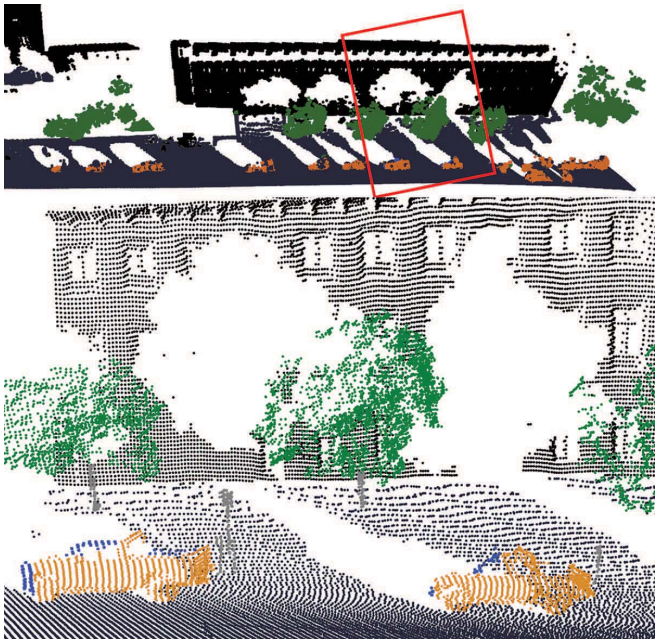


Fig. 11. Test result on the Oakland data.

observed that with the chosen 0.1m voxel resolution, the separation of the different MLS scene regions has been more efficient than by using the considered reference techniques (see Fig. 9(b)(d)(f)(h)(j)(l)). The last row of Fig. 9 demonstrate two failure cases of C^{2}CNN: the head of a traffic sign occluded by a tree crown cannot be detected (Fig. 9(m)), and a street column surrounded by multiple people cannot be separated from the pedestrian regions (Fig. 9(p)).

Note that the obtained voxel classification results can be further improved by simple spatial operations such as median filtering applied on local voxel neighborhoods. In addition, some false alarms may be removed in post processing, by adopting prior shape, size and location constraints for the detected object regions, upon available top-down knowledge about the scene.

E. Implementation Details and Running Time

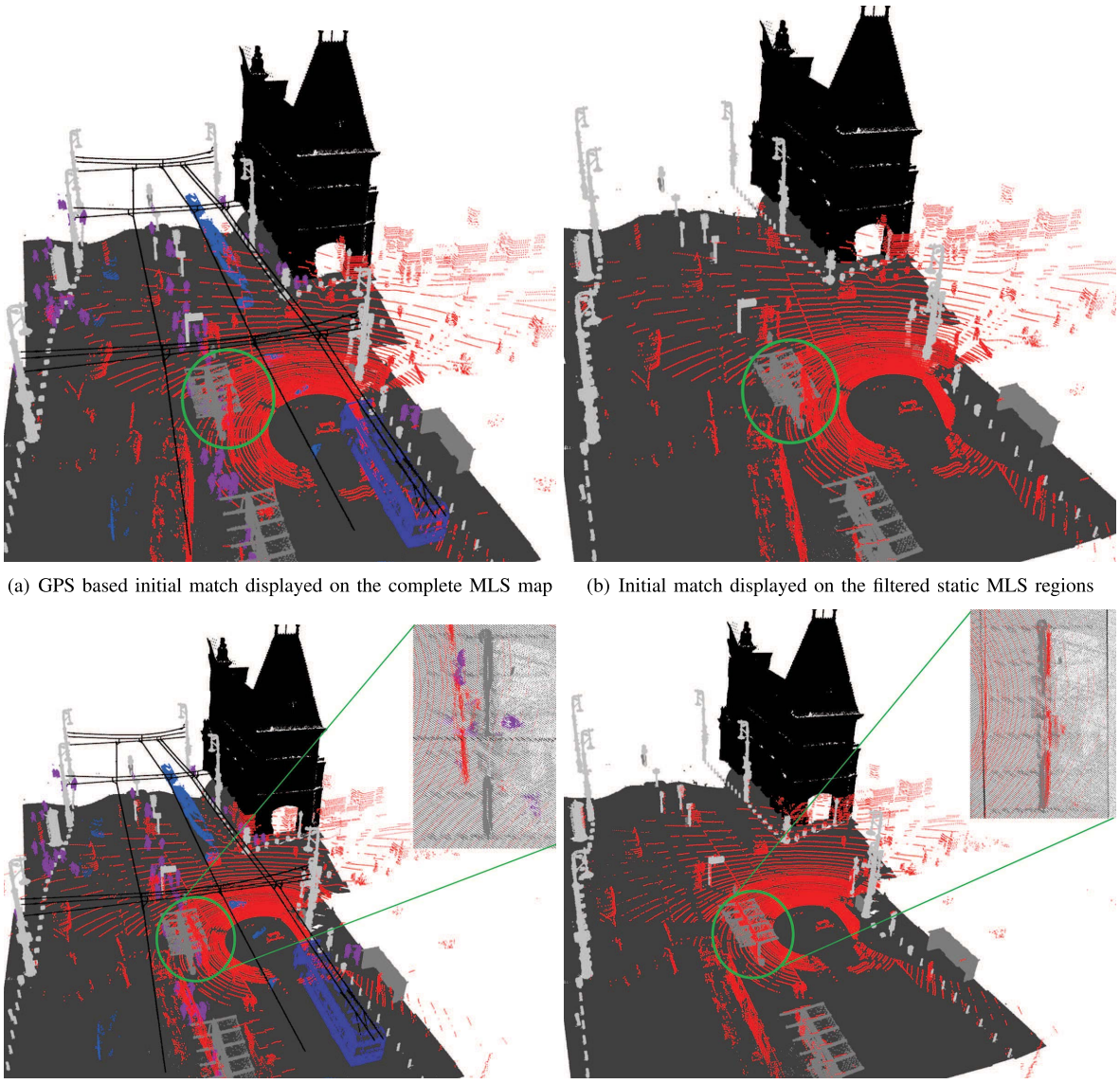
We implemented our training pipeline in Python using Keras and Tensorflow backend, while the further algorithmic modules were developed in C++ using OpenGL. Training the C^{2}CNN on the `SZTAKI CityMLS` dataset took around 36 hours, using a Nvidia Geforce GTX 1080 GPU with 8GB device and 64GB main memory. The label prediction step takes less than 10^{-4} seconds for a 2-channel $23 \times 23 \times 23$ training volume. As an example, by processing a complete scene with ground area 56 $m \times 111m$, 19M included points and $0.1m$ voxel resolution, our sparse voxel based space representation yielded 1.75M voxels, thus the overall label prediction took around 3 minutes.

We have measured the prediction time of the PointNet++, SPLATNet$_{3D}$ and the proposed C^{2}CNN techniques on a selected test scene containing 25 million points. We have experienced that while the PointNet++ showed the highest time complexity (563 sec), the running time of SPLATNet$_{3D}$ (198 sec) and the proposed approach (153 sec) proved to be notably shorter. As for the time complexity of the training step, our network is significantly quicker than the two reference methods due to its smaller structure.

F. Case Study on Vehicle Localization Based on the Semantically Labeled MLS Point Cloud

In this section, we provide a proof-of-concept validation for the efficiency of the proposed semantic point cloud classification approach in an automotive application. The discussed task is real-time decimeter-accurate localization, and orientation estimation of a self-driving vehicle (SDV) in the MLS point cloud map of the city, using the measurements of a rotating-multi-beam (RMB) Lidar sensor mounted on the top of the SDV [26]. In dense urban areas, such as the downtown of Budapest, Hungary, one can experience that commercial Global Positioning Systems (GPS) can often only provide position information with large inaccuracies (1-10 m). In order to correct the error of the initial GPS based transformation we proposed a low-cost method [1], [27] which is able to precisely register the sparse and inhomogeneous RMB Lidar point clouds of the SDV to a dense geo-referenced point cloud which can be obtained by a MLS mapping system: To reduce the complexity of the algorithm first we fit a rectangular 2D grid onto the horizontal plane and we project all the 3D points to the corresponding 2D grid cell. In the next step the blobs of the estimated obstacles are separated both in the sparse RMB Lidar point clouds and in the dense MLS map, using an adaptive connected component extraction algorithm applying structure-based *merge* and *split* steps. The core of the registration method is a quick object-level transformation estimation algorithm between the point clouds in the Hough domain, relying on several automatically extracted feature points [1]. Contrary to point level point cloud registration techniques such as the Iterative Closest Point (ICP) or the Normal Distributions Transform (NDT) [27], the proposed method [1] works in real-time, and it is able to manage arbitrary initial rotation differences between the two point clouds, as well as an initial translation error up to $10-15$m.

However, the above registration algorithm is based on the assumption that the reference landmark objects extracted from the MLS map correspond in majority to static and permanent



(a) GPS based initial match displayed on the complete MLS map

(b) Initial match displayed on the filtered static MLS regions

(c) Registration result [1] based on the complete map

(d) Registration result [1] on C^{2}CNN filtered map

Fig. 12. Application of the proposed C^{2}CNN classification approach for point cloud registration enhancement. Automatic registration results of a sparse RMB Lidar point cloud (shown with red in all images), to the dense MLS measurements (remaining colors). Figure (c) shows the registration results on the raw point cloud with notable inaccuracies (different class colors in MLS only serve better visibility). Figure (d) demonstrates the output of successful registration based on removing the dynamic objects from the MLS point cloud using the proposed C^{2}CNN method.

scene elements (such as lamp posts, tree trunks, kiosks etc), while all the phantoms, and moving or movable objects of the MLS point cloud appear as noise factors during the estimation of the right transform. Obviously, all scanning artifact have a great effect on the object assignment step, e.g. erroneously matching several *phantoms* in the MLS maps to static or dynamic objects of the RMB Lidar frames may increase the evidence of false global transforms in the Hough space. For this reason, semantic preliminary labeling of the MLS reference map is a critical step for this application.

Fig. 12 demonstrates the improvements on the registration results by exploiting the labels obtained by the proposed C^{2}CNN approach. In all subfigures, the RMB Lidar point cloud of the SDV is shown in *red*, while the MLS point cloud is displayed with the remaining different colors corresponding to the obtained C^{2}CNN-labels. The top row shows the purely GPS based alignment of the two point clouds, the only difference is that while Fig. 12(a) displays all points of the original MLS data, Fig. 12(b) contains the filtered static MLS regions only. We can observe here initial translation and rotation errors of around 7 meters and 8.5 degrees, respectively. The bottom row visualizes the registration results. In case of Fig. 12(c) the complete MLS point cloud was used as input of the registration [1], yielding notable inaccuracies. On the other hand, if we eliminate all phantoms and movable objects from the MLS map with C^{2}CNN, the registration process provides a successful output as shown in Fig. 12(d).

We also measured the advantages of the C^{2}CNN filter on registration accuracy in a quantitative way. We run the same registration algorithm [1] between the actual RMB Lidar frame

TABLE V

QUANTITATIVE EVALUATION OF THE POINT CLOUD REGISTRATION TECHNIQUE [1] BASED ON THE RAW MLS POINT CLOUD, THE SEMANTICALLY LABELED CLOUD USING THE PROPOSED C^{2}CNN APPROACH, AND THE MANUALLY LABELED DATA, RESPECTIVELY

Dataset	Raw MLS point cloud		C^{2}CNN labeled data		Manually labeled data	
	s [m]	rot [deg]	s [m]	rot [deg]	s [m]	rot [deg]
Main roads	1.74	3.92	0.37	1.19	0.26	0.97
Narrow roads	1.37	2.38	0.29	0.83	0.18	0.78
Crossroads	2.42	4.02	0.45	1.33	0.29	0.89
Small #of phantoms	0.93	1.60	0.26	0.87	0.21	0.77
Large #of phantoms	2.14	3.53	0.48	1.37	0.28	0.95
Overall	1.72	3.09	0.37	1.18	0.24	0.87

Note: Translation distance error (s) is given in meter and rotation error is given in degree.

and the MLS scenes in three configurations, using as reference map (i) the raw MLS point cloud, (ii) the static point cloud filtered by C^{2}CNN, and the (iii) manually filtered static point cloud. We divided the point cloud scenes for registration evaluation into different tests set: based on location category we distinguished narrow streets, main roads and large crossroads, while we also separated MLS scenes with dense and sparse phantom effects, respectively. The resulting registration errors in offset (s) and rotation (rot) are shown in Table V. We can see that the C^{2}CNN-based semantic filtering process significantly decreased the registration inaccuracies compared to the raw MLS data input, and the result's accuracy is very close to the output got by the utilization of the manually filtered map. The improvements are particularly significant in main roads and crossroads, where the presence of false landmarks is stronger without semantic labeling, and in the selected scenes with large phantom regions which could mislead the matching step working on raw data.

VI. CONCLUSION

We proposed a new 2-channel 3D CNN based technique to segment point cloud scenes obtained by Mobile Laser Scanning into nine different classes relevant for 3D High Definition city map generation. We have validated the efficiency of the approach in diverse and real test data from various urban environments, and demonstrated its advantages versus three baseline approaches. Additionally we introduced a potential real-life application for car self-localization of the proposed method. The authors would like to thank Budapest Közút Zrt for the provision of MLS test data.

REFERENCES

[1] B. Nagy and C. Benedek, "Real-time point cloud alignment for vehicle localization in a high resolution 3D map," in *Proc. Eur. Conf. Comput. Vis.*, Munich, Germany, 2018, pp. 226–239.

[2] H. Zheng, R. Wang, and S. Xu, "Recognizing street lighting poles from mobile LiDAR data," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 1, pp. 407–420, Jan. 2017.

[3] Y. Yu, J. Li, H. Guan, and C. Wang, "Automated detection of three-dimensional cars in mobile laser scanning point clouds using DBM-Hough-forests," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 7, pp. 4130–4142, Jul. 2016.

[4] B. Wu *et al.*, "A voxel-based method for automated identification and morphological parameters estimation of individual street trees from mobile laser scanning data," *Remote Sens.*, vol. 5, no. 2, pp. 584–611, 2013.

[5] B. Nagy and C. Benedek, "3D CNN based phantom object removing from mobile laser scanning data," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Anchorage, AK, USA, May 2017, pp. 4429–4435.

[6] G. Riegler, A. O. Ulusoy, and A. Geiger, "Octnet: Learning deep 3D representations at high resolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3577–3586.

[7] C. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, Long Beach, CA, USA, 2017, pp. 5105–5114.

[8] S. Papadimitriou, H. Kitagawa, P. B. Gibbons, and C. Faloutsos, "LOCI: Fast outlier detection using the local correlation integral," in *Proc. 19th Int. Conf. Data Eng.*, Los Alamitos, CA, USA, Mar. 2003, pp. 315–326.

[9] S. Sotoodeh, "Outlier detection in laser scanner point clouds," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 36, no. 5, pp. 297–302, 2006.

[10] J. Köhler, T. Nöll, G. Reis, and D. Stricker, "Robust outlier removal from point clouds acquired with structured light," in *Proc. Eurographics*, Cagliari, Italy, 2012, pp. 21–24.

[11] T. Kanzok, F. Süß, L. Linsen, and R. Rosenthal, "Efficient removal of inconsistencies in large multi-scan point clouds," in *Proc. Int. Conf. Central Eur. Comput. Graph., Visualizat. Comput. Vis.*, 2013, pp. 1–10.

[12] J. Gehrung, M. Hebel, M. Arens, and U. Stilla, "An approach to extract moving objects from MLS data using a volumetric background representation," *ISPRS Ann. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 4, p. 107, May 2017.

[13] A. Börcs, B. Nagy, and C. Benedek, "Instant object detection in lidar point clouds," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 7, pp. 992–996, Jul. 2017.

[14] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner, "Vote3Deep: Fast object detection in 3D point clouds using efficient convolutional neural networks," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, Singapore, May/Jun. 2017, pp. 1355–1361.

[15] J. Huang and S. You, "Point cloud labeling using 3D convolutional neural network," in *Proc. 23rd Int. Conf. Pattern Recognit. (ICPR)*, Cancun, Mexico, Dec. 2016, pp. 2670–2675.

[16] H. S. Koppula, A. Anand, T. Joachims, and A. Saxena, "Semantic labeling of 3D point clouds for indoor scenes," in *Proc. 24th Int. Conf. Neural Inf. Process. Syst.*, Granada, Spain, 2011, pp. 244–252.

[17] T. Hackel, J. D. Wegner, and K. Schindler, "Fast semantic segmentation of 3D point clouds with strongly varying density," *ISPRS Ann. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 3, no. 3, pp. 177–184, 2016.

[18] G. Pang and U. Neumann, "3D point cloud object detection with multi-view convolutional neural network," in *Proc. 23rd Int. Conf. Pattern Recognit. (ICPR)*, Cancun, Mexico, Dec. 2016, pp. 585–590.

[19] W. Wang, R. Yu, Q. Huang, and U. Neumann, "SGPN: Similarity group proposal network for 3D point cloud instance segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2569–2578.

[20] H. Su *et al.*, "SPLATNet: Sparse lattice networks for point cloud processing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2530–2539.

[21] D. Munoz, J. A. Bagnell, N. Vandapel, and M. Hebert, "Contextual classification with functional max-margin Markov networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 975–982.

[22] A. Serna, B. Marcotegui, F. Goulette, and J. Deschaud, "Paris-rue-Madame database: A 3D mobile laser scanner dataset for benchmarking urban detection, segmentation and classification methods," in *Proc. 4th Int. Conf. Pattern Recognit., Appl. Methods (ICPRAM)*, Angers, France, Mar. 2014, pp. 1–6.

[23] B. Vallet, M. Brédif, A. Serna, B. Marcotegui, and N. Paparoditis, "TerraMobilita/iQmulus urban point cloud analysis benchmark," *Comput. Graph.*, vol. 49, pp. 126–133, Jun. 2015.

[24] T. Hackel, N. Savinov, L. Ladicky, J. Wegner, K. Schindler, and M. Pollefeys, "Semantic3D.net: A new large-scale point cloud classification benchmark," *ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci.*, vol. 4, pp. 91–98, May 2017.
[25] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[26] A. Börcs, B. Nagy, and C. Benedek, "Fast 3-D Urban object detection on streaming point clouds," in *Proc. Eur. Conf. Comput. Vis.*, 2015, pp. 628–639.
[27] B. Gálai, B. Nagy, and C. Benedek, "Crossmodal point cloud registration in the Hough space for mobile laser scanning data," in *Proc. 23rd Int. Conf. Pattern Recognit. (ICPR)*, Cancun, Mexico, Dec. 2016, pp. 3374–3379.

Balázs Nagy received the M.Sc. degree in computer engineering from Peter Pazmany Catholic University (PPCU), in 2016, where he is currently pursuing the Ph.D. degree.

His research interests include scene segmentation and object recognition from 3D point cloud data using geometric and machine learning approaches.

He is also a member of the Machine Perception Research Laboratory, Institute for Computer Science and Control, Hungarian Academy of Sciences (MTA SZTAKI), where he has been a key Researcher and Developer in many projects related to environment analysis and self-driving vehicles.

Csaba Benedek received the M.Sc. degree in computer sciences from the Budapest University of Technology and Economics (BME), in 2004, and the Ph.D. degree in image processing from Péter Pázmány Catholic University, Budapest, in 2008.

From 2008 to 2009, he was a Post-Doctoral Researcher with INRIA, Sophia-Antipolis, France. He has been the manager of various national and international research projects in recent years. He is currently a Senior Research Fellow with the Machine Perception Research Laboratory, Institute for Computer Science and Control of the Hungarian Academy of Sciences (MTA SZTAKI), and a Habilitated Associate Professor with Péter Pázmány Catholic University. His research interests include Bayesian image and point cloud segmentation, object extraction, change detection, and GIS data analysis.