

# Urban 3D segmentation and modelling from street view images and LiDAR point clouds

Pouria Babahajiani<sup>1</sup>  · Lixin Fan<sup>1</sup> · Joni-Kristian Kämäräinen<sup>2</sup> · Moncef Gabbouj<sup>2</sup>

Received: 4 September 2016 / Revised: 23 March 2017 / Accepted: 26 April 2017 / Published online: 29 May 2017  
© The Author(s) 2017. This article is an open access publication

**Abstract** 3D urban maps with semantic labels and metric information are not only essential for the next generation robots such as autonomous vehicles and city drones, but also help to visualize and augment local environment in mobile user applications. The machine vision challenge is to generate accurate urban maps from existing data with minimal manual annotation. In this work, we propose a novel methodology that takes GPS registered LiDAR (Light Detection And Ranging) point clouds and street view images as inputs and creates semantic labels for the 3D points clouds using a hybrid of rule-based parsing and learning-based labelling that combine point cloud and photometric features. The rule-based parsing boosts segmentation of simple and large structures such as street surfaces and building facades that span almost 75% of the point cloud data. For more complex structures, such as cars, trees and pedestrians, we adopt boosted decision trees that exploit both structure (LiDAR) and photometric (street view) features. We provide qualitative examples of our methodology in 3D visualization where we construct parametric graphical models from labelled data and in 2D image segmentation where 3D labels are back projected to the street view images. In quantitative evaluation we report classification accuracy and computing times

and compare results to competing methods with three popular databases: NAVTEQ True, Paris-Rue-Madame and TLS (terrestrial laser scanned) Velodyne.

**Keywords** Urban 3D · Point cloud · LiDAR · Street view · Semantic segmentation · Robotics

## 1 Introduction

3D urban map model is a digital representation of the earth's surface at city locations consisting of terrestrial objects such as buildings, trees, vegetation and manmade objects belonging to the city area. 3D maps are useful in different applications such as architecture and civil engineering, virtual and augmented reality, and modern robotics (autonomous cars and city drones). Creating photorealistic and accurate 3D urban maps requires high volume and expensive data collection. For example, Google and Nokia HERE have cars mounted with cameras and Light Detection And Ranging (LiDAR) scanners to capture 3D point cloud and street view data along streets throughout the world. While laser scanning or LiDAR systems provide a readily available solution for capturing spatial data in a fast, efficient and highly accurate way, the semantic labelling of data would require enormous man power if done manually. Therefore, the problem of automatic labelling (parsing) of 3D urban data to associate each 3D point with a semantic class label (such as “car”, “tree”) has gained momentum in the computer vision community [11, 17, 24, 42].

Automatic segmentation and labelling of urban point cloud data is challenging due to a number of data specific challenges. First, high-end laser scanning devices output millions of data points per second, and therefore the methods need to be efficient to cope with the sheer volume of the

✉ Pouria Babahajiani  
pouria.babahajiani.ext@nokia.com

Lixin Fan  
Lixin.Fan@nokia.com

Joni-Kristian Kämäräinen  
joni.kamarainen@tut.fi

Moncef Gabbouj  
Moncef.Gabbouj@tut.fi

<sup>1</sup> Nokia Technologies, Tampere, Finland

<sup>2</sup> Department of Signal Processing, Tampere University of Technology, Tampere, Finland

urban scene datasets. Second, point cloud sub-regions corresponding to individual objects are imbalanced, varying from sparse representations of distant objects to dense clouds of nearby objects, and incomplete (only one side of objects is scanned by LiDAR). Third, for accurate object recognition a sufficiently large labelled training data (ground truth) are needed to train the best supervised methods.

In this work, we tackle the efficiency issue by proposing a hybrid method which consists of following three steps: First, certain simple but frequently occurring structures, such as building facades and ground surface, are quickly segmented by rule-based methods. The rule-based method can typically label 70–80% of the point cloud data and rule-based methods are more than  $6\times$  faster than the otherwise efficient boosted decision trees [17]. Second, the remaining points are processed with our fast supervised classifier. To construct high-quality features for our classifier, we first over-segment the points to 3D voxels which are further joined into super-voxels from which structure features are extracted. Moreover, as the 3D points are aligned with street view images we also extract photometric features. Our classifier of choice is a boosted decision tree classifier which is trained to label the remaining points using the super-voxel features. Third, We solve the problem of incomplete data by utilizing parametric 3D templates of certain classes (cars, trees and pedestrians) and fit them to the boosted decision tree labelled super-voxel point clouds. The final step also improves the visual quality of the semantic 3D models output from our processing pipeline, especially for those sparse and incomplete point clouds corresponding to small objects. Another application of our method is semantic segmentation of street view images which is achieved by backprojecting the semantic labels of the point cloud points to the corresponding street view images. Figure 1 depicts the overall workflow of our method. We provide qualitative examples of 3D visualization and 2D segmentation and in quantitative experiments we report and compare our segmentation accuracy and computing time to previous works. This work is based on the preliminary results in [2,3], but provides a significant extension since it contains experimental results on three publicly available datasets, comparison to other recent works, refined processing steps and an extensive ablation study over the method parameters.

**Contributions** Preliminary results on components of our processing pipeline have been reported in [2,3], and in this work we make the following novel contributions:

- We have demonstrated a complete urban map data processing pipeline, which annotate all 3D LiDAR points with semantic labels. Our method is made efficient by combining fast rule-based processing for building and street surface segmentation and super-voxel-based feature extraction and classification for remaining map elements (cars, pedestrians, trees and traffic signs).
- We propose two back ends for semantically labelled urban 3D map data that exemplify two important applications: (i) 3D urban map visualization and (ii) semantic segmentation of 2D street view images by backprojection of the 3D labels.
- Parameters of the different processing stages have clear physical and intuitive meaning, and therefore they are easy to set for novel data or optimize by cross-validation over certain ranges. We have made extensive experiments on larger datasets, and moreover, optimal parameter settings are cross-validated against labelled datasets. Experimental results verify superior accuracy and efficiency of our method as compared to the existing works on three difficult datasets.

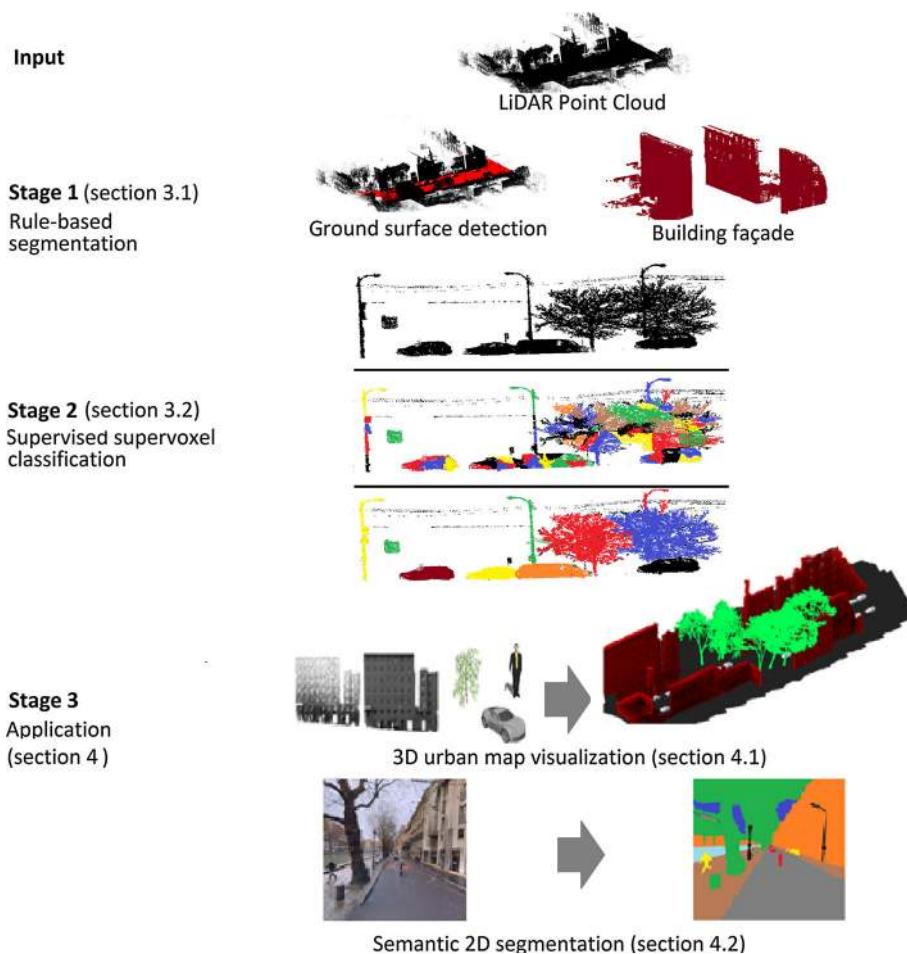
As such we provide full processing pipeline from 3D LiDAR point cloud and street view image data (cf. Google Maps and Nokia HERE) to urban 3D map data visualization and to 2D semantic segmentation. All parameters have physical meaning, and the system automatically adapts to the dataset size.

## 2 Related work

3D segmentation and labelling (classification) using image and point cloud data of urban environments have many potential applications in augmented reality and robotics and therefore research on these topics has gained momentum during the last few years. In the following, we briefly mention the most important 2D approaches, but focus on 3D point cloud methods and methods particularly tailored for urban 3D processing. Several important surveys have been recently published where more details of specific approaches can be found [26,27,37].

**Image-based methods** Due to the lack of affordable and high-quality 3D sensors until the introduction of Kinect in 2010 the vast majority of the works is still based on 2D (RGB) image processing. Progress in 2D over the years has been remarkable and for 2D object classification and detection there have been several breakthrough methodologies [23,45], in particular, the visual Bag-of-Words (BoW) [6,35], Scale Invariant Feature Transform (SIFT) [46] and the Deformable Part Model (DPM) [9]. Recently, these methods have been superseded by methods using deep convolutional neural networks, e.g. AlexNet [19] and R-CNN [10]. Direct applicability of these methods is unclear since the datasets used in training contain objects in various non-urban environments (ImageNet, Pascal VOC) and sources of detection failures may therefore be different. The deep neural network methods also require large annotated datasets. Moreover, mapping the 2D

**Fig. 1** The overall workflow of the proposed methodology



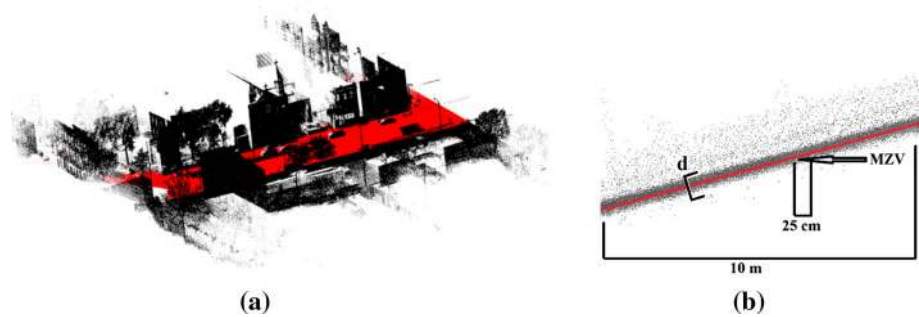
bounding boxes to 3D point cloud object boundaries is not trivial. To summarize, state-of-the-art 2D methods provide a potential research direction as combined with state-of-the-art 3D methods, but in this work we focus on methods particularly developed for urban 3D map data segmentation.

*Point cloud-based methods* The success of local descriptors in 2D has inspired to develop 3D local descriptors for point cloud data, e.g. 3D SURF (speeded up robust features) [18] and 3D HOG (histogram of oriented gradients) and DoG (difference of Gaussians) [43], and their comparison can be found from the two recent surveys [13, 14]. These methods and also many direct point cloud-based methods, e.g. [4, 15, 28], are designed to recognize a specific object stored as a point cloud model, and therefore practical use of these methods for urban 3D often requires various geometric features to robustify matching [38].

*Urban 3D segmentation* Most of the existing city modelling approaches directly or indirectly tackle the problem through 3D point cloud analysis. Lafarge and Mallet [20] applied a Markov Random Field (MRF) -based on optimization technique, using the graph-cut framework for object detection using airborne laser scanner (ALS) data. In this work, we omit 3D data generated by airborne devices (see, e.g. [12, 20]) and

assume that the 3D map data have been collected via terrestrial and mobile laser scanning—this kind of data is available, for example, in Google Maps and Nokia HERE maps where the 3D data are mobile laser scanner (MLS) LiDAR generated point cloud. It is noteworthy that urban 3D segmentation has also been investigated for stereo-generated point clouds [32], but there noise level is orders of magnitude higher and therefore we focus on high-quality LiDAR data. Douillard et al. [8] compared various segmentation approaches for dense and sparse LiDAR data and found simple clustering methods the best and noted that street pre-segmentation improves the results. These findings were verified in the survey by Nguyen and Le [27] who also pointed that learning-based methods are needed for more complex objects due to noise, uneven density and occlusions. Inspired by these two important findings, we adopt a fast rule-based approach for simple and frequently appearing structures (streets and building facades) and the learning-based approach for more complex structures. The combination of clustering, extracting geometric features and using a supervised classifier to recognize objects was proposed in [11], but in our approach we accelerate computation by the rule-based pre-processing and by adopting the efficient super-voxel clustering that has been used in video process-

**Fig. 2** Example of rule-based segmentation of road surfaces. **a** 3D LiDAR point cloud segmented to *road surface points* (red) and *other points* (black); **b** a sketch illustrating our plane fitting to one tile



ing [41]. Fast 3D-only methods exist [24], but it has also been argued that joint 2D image cues (e.g. colour, texture, shape) and 3D information (point cloud) provide higher accuracy [40,42,44] and therefore we collect features from 3D and 2D for our classifier.

### 3 Proposed methodology

The overall processing steps of our approach are illustrated in Fig. 1. The input to our processing algorithm is 3D LiDAR point cloud  $\mathcal{P} = \{\mathbf{p}_i\}$  ( $\mathbf{p} \in \mathbb{R}^3$ ) and street view images  $\mathcal{I} = \{\mathbf{I}_i\}$  ( $\mathbf{I} \in \mathbb{R}^{W \times H \times 3}$ ). The camera and LiDAR sensors are calibrated with respect to each other. The first processing step of our methodology is the rule-based segmentation of the ground surface (Sect. 3.1.1) and building facades (Sect. 3.1.2). The points labelled by the rule-based processing cover approximately 75% of the urban city data, and the remaining points proceed to the next step. The next step is super-voxel clustering (Sect. 3.2.1) and feature extraction from each super-voxel after which the super-voxels are classified using the boosted decision tree classifier (Sect. 3.2.2). The output of the method is a fully labelled 3D point cloud where each point is labelled to belong to one of the pre-defined semantic classes (Fig. 1). We also present two different applications of our system: (i) 3D urban map visualization (Sect. 4.1) by using parametric models generated from the labelled super-voxels and (ii) 2D segmentation (Sect. 4.2) by mapping the 3D labels to the RGB images.

#### 3.1 Rule-based segmentation of simple structures

Our empirical findings are in align with [8,27] which pointed clear computational advances for using pre-processing to segment geometrically simple and dominating structures. Therefore, we devise simple rule-based detectors for *road surfaces* and *building facades* that span majority of urban scene point clouds (75% on average in the datasets used in the experiments). Both road surfaces and building facades form large and dense horizontal and vertical planar regions, and therefore it is easy to devise geometric rules constraining them and providing fast segmentation as compared to

the learning-based approaches. The rule-based steps detect and label the road surface points  $\mathcal{P}_{\text{road}}$ , and building points  $\mathcal{P}_{\text{building}}$  which are removed from the original point cloud  $\mathcal{P}' = \mathcal{P} \setminus (\mathcal{P}_{\text{road}} \cup \mathcal{P}_{\text{building}})$  and then passed to the next processing step (learning-based segmentation).

##### 3.1.1 Road surfaces

The goal of the first step is to detect road surface points including the car path and side-walk, and as a result, the original point cloud is divided into *road surface* ( $\mathcal{P}_{\text{road}}$ ) and other ( $\mathcal{P}_{\text{other}} = \mathcal{P} \setminus \mathcal{P}_{\text{road}}$ ) points (Fig. 2). Starting from the road surfaces is also beneficial for the later steps that are based on point cloud connectivity as the road and ground surfaces connect almost all points together. We apply the fast and robust Random sample consensus (RANSAC)-based plane fitting similar to Lai and Fox [21] who used it to remove planar regions from Google 3D Warehouse point clouds.

To adapt the Lai and Fox algorithm for our case of large urban city maps we need to do two additional steps: (i) local fitting and (ii) windowed candidate surface point selection. The first step is needed to allow varying street slope (cf. ‘‘San Francisco’’ landscape). The second step is needed to decrease the total number of points for plane fitting to make it faster.

Therefore, our adapted algorithm consists of following three steps: First, the original point cloud is divided into smaller point clouds  $\{\mathcal{P}_{10\text{ m} \times 10\text{ m}}\}_k$  which span  $10 \times 10$  m square areas. Secondly, each  $\{\mathcal{P}_{10\text{ m} \times 10\text{ m}}\}_k$  point cloud is further divided into  $0.25\text{ m} \times 0.25\text{ m}$  cells, and for each cell the Minimal-Z-Value (MZV) is computed by averaging 10 lowest z-value points.<sup>1</sup> Thirdly, for each cell, all points lying within  $\pm\tau_{\text{MZV}}$  distance from MZV are selected for plane fitting (Fig. 2). The selection process reduces the number of points to around 0.1% from the original, and in all experiments we fixed the threshold to  $\tau_{\text{MZV}} = 0.02\text{ m}$ . For each local point cloud the points that are within the distance  $\tau_{\text{road}} = \pm 0.08\text{ m}$  from the fitted plane are added to  $\mathcal{P}_{\text{road}}$ . The average processing time of a single  $10\text{ m} \times 10\text{ m}$  region is about 15 ms. This approach is not sensitive to the selec-

<sup>1</sup> Using 10 lowest z-value points is to make the MZV estimation insensitive to outliers.

tion of the two thresholds and efficiently segments the road surface points.

### 3.1.2 Building facades

The workflow of our rule-based building facade segmentation is shown in Fig. 3. At first, we form a GPS-defined x-y plane similar to the road surface detection and divide the plane to 0.25 m × 0.25 m cells. Our detection rules are derived from the dominant characteristics of building facades in LiDAR point clouds: LiDAR provides high (z dimension) and dense regions. Since the x-y plane is now divided to the discrete cells,  $\Delta_x, \Delta_y$ , we can compute height and density features. We use proportional measures that make them invariant to the average height of a city (e.g. Paris vs. New York City). As a height feature we use

$$P_h(\Delta_x, \Delta_y) = \frac{\operatorname{argmax}_z P(\Delta_x, \Delta_y, z)}{\operatorname{argmax}_z p(:, :, z)}, \tag{1}$$

and as a density feature

$$P_d(\Delta_x, \Delta_y) = \frac{|P(\Delta_x, \Delta_y, z)|}{\max_{\Delta_x, \Delta_y} |P(\Delta_x, \Delta_y, z)|}. \tag{2}$$

Equations. (1) and (2) are combined to the final “building score”:

$$P_{\text{building}}(\Delta_x, \Delta_y) = P_h(\Delta_x, \Delta_y) + \lambda_d P_d(\Delta_x, \Delta_y). \tag{3}$$

In our experiments we used simple maximum heights, but more robust score can be constructed by adopting robust statistics (rank-order statistics) where the maximum value is replaced, e.g. by the value that is higher than 95% of all points. The maximum value performed well in our experiments and we fixed the balancing factor  $\lambda_d = 1.0$  for equal weighting for the height and density scores. Moreover, this approach is insensitive to the number of points to compute the score number. In our experiments a notable speed up

is attained without significant loss in accuracy, in the case that up to 30% of input points have been randomly removed from computation. The rule-based segmentation of buildings is achieved by computing the building score in (3) to the cells of size 0.25 m × 0.25 m (the same as before) and thresholding each cell by  $\tau_{\text{building}} = 1.80$ . This generates a binary x-y map (Fig. 3) for which we compute the standard shape compactness features for each connected shape  $S_i$  [22]:

$$P(S_i) = \frac{\pi \cdot \text{diameter}^2(S_i)}{4 \cdot \text{area}(S_i)}. \tag{4}$$

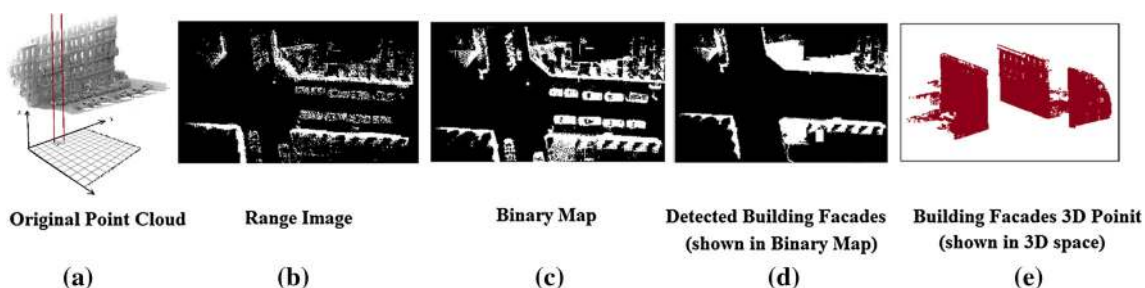
Again  $P(S_i)$  score is thresholded by  $\tau_{P(S_i)} = 15$  and the binary label as {building, ¬building} is backprojected to each 3D point within each cell. Note that this process is executed for a point set from which the street surface points have already been removed  $\mathcal{P}_{\text{other}}$  and this rule-based step creates another set  $\mathcal{P}'_{\text{other}} = \mathcal{P}_{\text{other}} \setminus \mathcal{P}_{\text{building}} = \mathcal{P} \setminus (\mathcal{P}_{\text{road}} \cup \mathcal{P}_{\text{building}})$ .

## 3.2 Boosted decision tree detector for super-voxel features

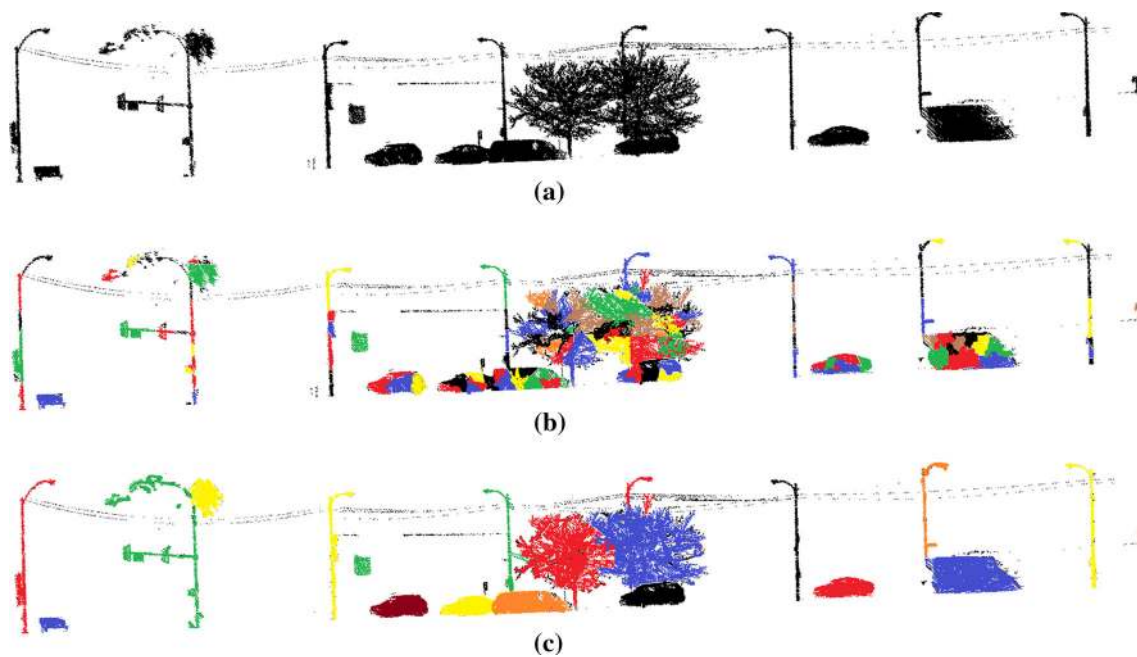
In our case, the number of 3D points is still large after the rule-based segmentation of roads and buildings and therefore we must consider both performance and efficiency issues for the supervised detection stage. In Fig. 4 is depicted the workflow of our supervised detection that processes the point cloud  $\mathcal{P}'_{\text{other}}$ . Our approach is inspired by the super-voxel-based processing successfully used in video analysis [41].

### 3.2.1 Super voxels

The first step is 3D point-wise agglomerative clustering that over-segments the input point cloud to voxels (Fig. 4b). The clustering algorithm incrementally picks a random seed points, adds points to the seed cluster to construct a voxel until no more points pass a distance-based merging rule and then pick a new seed point until all points have been processed. For the random seed point  $\mathcal{P}_i$  new points  $\mathbf{p}_j$  are added  $\mathcal{P}_i = \mathcal{P}_i \cup \mathbf{p}_j$  if they pass the distance rule,



**Fig. 3** Example of rule-based segmentation of building facades. **a** original 3D point cloud and GPS-define x-y plane for projection; **b** x-y projected points; **c** binary x-y map; **d** points that pass the building facade detection step; **e** backprojection of the detected points to 3D



**Fig. 4** The workflow of our supervised detection. **a** Input  $\mathcal{P}'_{other}$  point cloud where road surface and building facade 3D points have been removed; **b** point cloud over-segmentation to 3D voxels by agglomerative clustering; **c** super-voxelization by voxel-level agglomerative clustering

$$\min_i \text{dist}(\mathcal{P}_i, \mathcal{p}_j) \leq \tau_{\text{voxel}}, \quad (5)$$

where  $\text{dist}(\cdot, \cdot)$  is the minimal distance between a set and a point, and the distance threshold is set to  $\tau_{\text{voxel}} = 0.005$  m. After the first step, all points have been assigned to a single voxel. The distance threshold avoids setting the number of clusters which highly depends on the size of the point cloud and therefore metric threshold is more intuitive. We refer acute readers to Sect. 5.2 for ablation study concerning the setting of this crucial distance threshold.

The procedure of *super-voxelization* is to merge those voxels that are close to each other and share similar orientation. Formally, the proximity between two voxels  $\mathcal{P}_i$  and  $\mathcal{P}_j$  is defined as

$$\min_{i,j} \text{dist}(\mathcal{P}_i, \mathcal{P}_j) \leq \tau_{sv\_prox} \quad (6)$$

which is equivalent to the minimum-link distance rule in agglomerative clustering. The surface orientation is computed using the PCA method in [39] for each voxel and two voxels are combined if their normals are similar

$$\arccos(\text{norm}_{PCA}(\mathcal{P}_i), \text{norm}_{PCA}(\mathcal{P}_j)) \leq \tau_{sv\_orient}. \quad (7)$$

We set the super-voxelization thresholds to  $\tau_{sv\text{oxel}1} = 0.01$  m and  $\tau_{sv\text{oxel}2} = 15$  which produce high-quality super-voxels on all our datasets (see Fig. 4). The two thresholds with intuitive physical interpretation again avoid setting the number of clusters that would depend on the size of the point cloud.

### 3.2.2 Super-voxel classification

The automatically generated super-voxels can be classified by computing the popular 3D shape descriptors as features [13, 14], but we found these slow to compute and due to variance in point density their robust usage would require re-sampling which is a slow procedure as well. Instead, motivated by success of features with true physical meaning in voxelization and super-voxelization, we adopt several fast-to-compute physical measures as features. The selected features are listed in Table 1.

The features are fed to the boosted decision tree classifier [5] which is extremely efficient and produces high accuracy for multi-class classification tasks. The boosting is based on minimizing the exponential loss:

$$\sum_{i=1}^M \exp(-y_i f_\lambda(\mathbf{x}_i)) \quad (8)$$

where  $\mathbf{x}_i$  are the input features and  $y_i$  the ground-truth class labels and  $f_\lambda(\cdot)$  is the estimated label constructed from

$$f_\lambda(\mathbf{x}) = \sum_{j=1}^N \lambda_j h_j(\mathbf{x}) \quad (9)$$

where  $h_j(\cdot)$  is a weak learner and  $\lambda_j$  its corresponding weight parameter. Selection of the weak learners and optimization of the weights to minimize the loss function can be done efficiently by parallel updating which is faster than the

**Table 1** Simple geometric and photometric primitives used to classify super-voxels into pre-defined categories

Feature	Motivation
<i>Geometric features</i>	
Area	Small versus large objects
Edge ratio	Maximum and minimum edges
Max edge	Longest dimension
Covariance	overall shape
<i>Location and orientation features</i>	
Height above road	
Distance to street	Horizontal distance to the car GPS
Normal angle	With respect to the surface orientation
<i>3D and photometric texture features</i>	
Mean intensity	Overall reflection property of the voxel
Density	Density of the points
Planarity	Average distance to the best fitted plane

sequential-update algorithm [5], but we adopted the sequential version due to its simplicity and widespread availability. In the experiments, we used a forest of ten decision trees with each of them having six leaf nodes and this classifier leads to satisfactory classification results for the benchmark datasets used in our work.

## 4 Applications

The outputs of the two rule-based steps and the supervised detector based step are two large point clouds  $\mathcal{P}_{road}$  and  $\mathcal{P}_{building}$  and a number of smaller point clouds  $\mathcal{P}_i$  with assigned labels

$$l_{road}, l_{building}, l_i \in \{road, building, tree, car, pedestrian, \dots\}.$$

Using the point clouds, street view images and the labels we introduce two important applications: (1) enhanced 3D visualization using model-based rendering and (2) 2D semantic segmentation. In the first application we replace the annotated point clouds with 3D graphical models whose parameters are derived from the point cloud properties which provides visually more plausible view to the 3D map data. In the second application we back project the point cloud labels

to 2D street view images and demonstrate their usage in 2D semantic segmentation.

### 4.1 Visualization of 3D urban maps

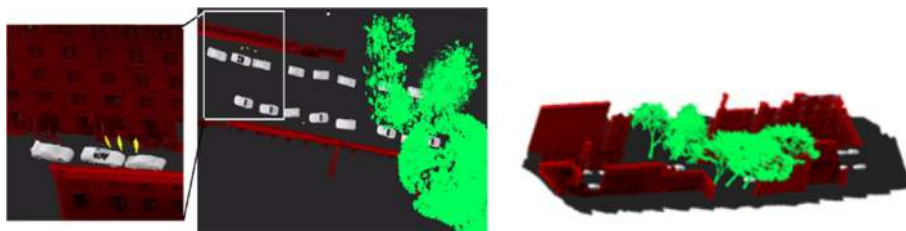
Our LiDAR point cloud and street view images are registered, i.e. the 3D projective transformations from the street view images to the point clouds are available, due to the common data acquisition by a data collection vehicle. A textured 3D model is typically generated by directly using image values or using parametric models [26]. Image RGB mapping is a fast procedure, but requires mesh generation as the pre-processing step which is time-consuming for large point clouds and is error-prone for noisy datasets. In this section, we introduce our fast rendering-friendly approach that reconstructs 3D urban map model in two stages (Fig. 5). Firstly we use the enhanced ShadVis algorithm [36] to fast render the building facades with high-quality details. The algorithm calculates the illumination of a point cloud with the light coming from a theoretical hemisphere or sphere around the object. In the second step we apply methods to fit pre-designed template models to non-building labelled point clouds.

#### 4.1.1 Building facade rendering

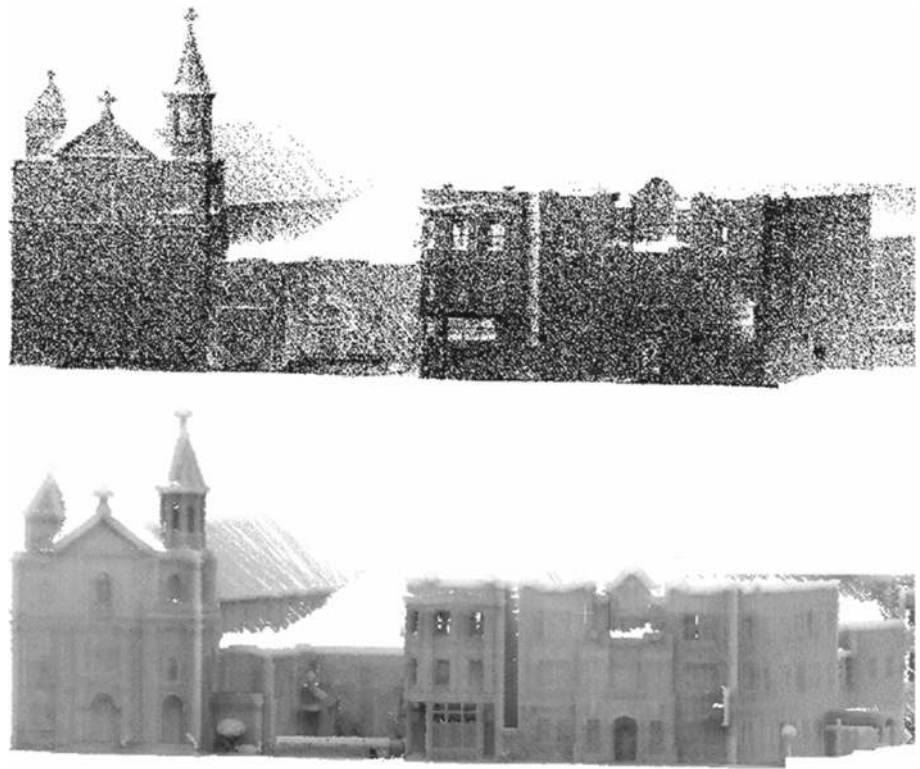
For fast rendering with a high level of details we apply the ShadVis technique in [36]. ShadVis estimates model illuminants as if the light was coming from a theoretical hemisphere or sphere around the object. The graphics hardware rendering pipelines have been designed for polygons, but in our case it is computationally more attractive to render only the points. Therefore, we adopt the simple but effective algorithm in [25]. The accuracy of the result depends on the resolution of the 3D point cloud dataset (see Fig. 6 for a typical case).

#### 4.1.2 Rendering object models

Buildings are large objects with sufficient number of 3D points for high-quality point-wise rendering, but this is not the case for small objects such as cars, trees and pedestrians. However, there are available numerous high-quality 3D models of many visual classes (e.g. 3D Warehouse <http://3dwarehouse.sketchup.com>) and these can be used to construct more plausible map view. The main problem in using

**Fig. 5** Example urban 3D map with rendered object models

**Fig. 6** Example of 3D points in  $\mathcal{P}_{building}$  (approx. 15 m distance, *top*) and results of our point rendering algorithm (*bottom*)



3D models is fitting model to a 3D point cloud. The fitting method is categorized into two types, depending on whether orientation of an object in question plays an important role in rendering (which is critical in our system where better visualization is the goal). The first type of fittings is related to the object classes which their pre-designed mesh structure orientation is not important and their object models will be based on their position and dimension only. The first object type fitting includes trees, pedestrians and sign symbols. Unlike a lot of work which calculate the distance of a given points to the closest surface and use time-consuming iterative procedure to fit the pre-designed model into the point cloud or reconstructed surface [7], we propose a novel approach to solve this problem in a straightforward and computationally lightweight manner. For each separated point cloud  $\mathcal{P}_i$ , the centre and its boundaries (3D bounding box) will be calculated. Based on the size of the existing pre-designed library meshes, we localize the best isodiametric meshes to the point cloud. Then, as the object orientation is not important we fit the mesh by stretching it to get an appropriate size. This is a similarity transformation of estimated isotropic scale  $s$  and transformation  $\mathbf{t} = (t_x, t_y, t_z)^T$ . It is also possible to estimate a similarity transformation where scale is applied to each dimension  $\mathbf{s} = (s_x, s_y, s_z)^T$ .

The second type object requires also x-y orientation angle  $\theta$  and is needed for different types of vehicles in our data (car, bus, bike). First, the centre of pre-designed mesh is computed and point cloud will be matched and then the corresponding

model will be chosen from library based on the dimension of the vehicle 3D bounding box. Then Iterative Closet Point (ICP) algorithm [30] is applied to automatically refine the registration of point clouds with desired mesh. The scene prior knowledge reduces the number of possible vehicle orientations as the road surface is determined (sect. 3.1.1) and only rotations around the z-axis of the road are considered. The ICP algorithm that we apply optimizes the RMS (Root mean Square) distance between closest point pairs of the models vertices to the point cloud [30]

$$err_{RMS}(\mathcal{P}_{model}, \mathcal{P}_i) = \sqrt{\frac{1}{N} \sum_{n=1}^N \| \mathbf{p}_{n,model} - \mathbf{p}_{n,\mathcal{P}_i} \|^2} \quad (10)$$

In Fig. 7 is illustrated model fitting to a point cloud. Notice that even with a different target model of the car (sedan vs. hatchback) correct pose is readily estimated.

#### 4.2 Semantic segmentation in 2D

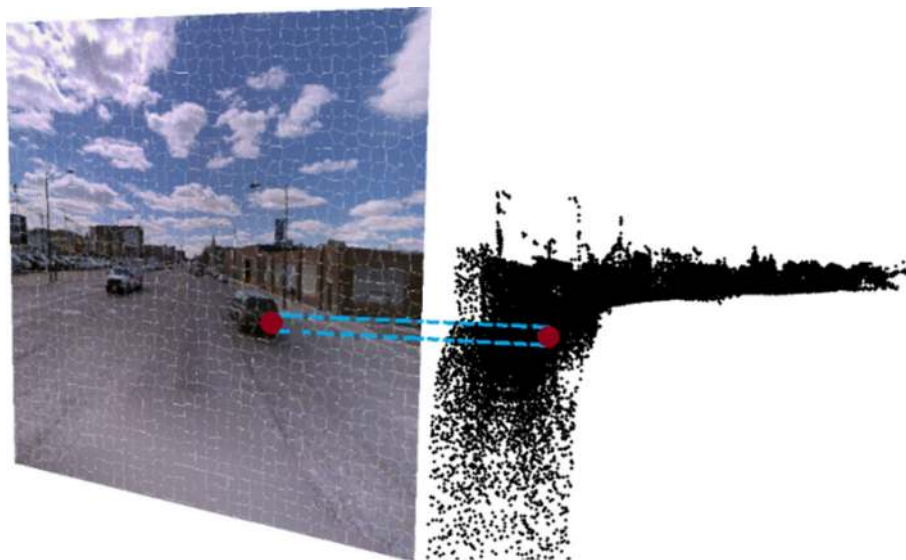
Thanks to the Global Positioning System (GPS) and Inertial Measurement Unit (IMU) measurements in the mobile LiDAR and RGB data acquisition system there is accurate information to register the 3D point cloud and street view (RGB) data (Fig. 8). Therefore, it is straightforward to map the semantic labels of 3D point cloud points to the street view images. For computational efficiency, input images





**Fig. 7** Example of a 3D car model fitted and rendered to a point cloud

**Fig. 8** Mapping between the 3D LiDAR points and 2D street view images



are over-segmented into super-pixels and each image plane super-pixel is associated with a collection of labelled 3D points (Fig. 8). For projection 3D points to image plane we use the generic camera model (images are already rectified to remove the optical distortions) [16]:

$$p_{img} = \mathbf{K} [\mathbf{R} | \mathbf{t}] p_{3d} \quad (11)$$

where  $\mathbf{t}$  is a  $3 \times 1$  translation vector,  $\mathbf{R}$  is a  $3 \times 3$  rotation matrix and  $\mathbf{K}$  is a  $3 \times 3$  camera matrix. The input and output data are given in the homogeneous coordinate system. All LiDAR points are transformed to each street view image and mapped to the closest super-pixel. The mapping uses z-buffering (within the same pixel only the closest 3D is selected), and majority vote label of 3D points projected to the same super-pixel is selected. The image super-pixels without any label are labelled as “sky”.

## 5 Experiments

In this section, we provide qualitative and quantitative results for the applications of visualization of urban 3D map data and semantic 2D segmentation. We compute the point-wise and

pixel-wise classification accuracies and compare our method to various recently proposed methods.

### 5.1 Datasets

*NAVTEQ True* the dataset used in this work is described in our previous work [2] and is composed of 500 high-quality street view images of  $1032 \times 1032$  resolution and corresponding LiDAR point clouds collected from three cities: Chicago, Paris and Helsinki. The data were collected using the NAVTEQ True systems of high-density 360° rotating LiDAR system, 360° panoramic camera and an inertial navigation system (IMU/GPS) for precise position, orientation and attitude tracking of the sensors. Information from all these sensors is synchronized to create an accurate and comprehensive dataset. The LiDAR system has 64 lasers and rotates at 600 rpm covering a full 360° field of view around the car. The LiDAR system scans 3D points at the rate of around 1.2 million points per second. NAVTEQ dataset is acquired in various weather conditions and urban landscapes and represents the most challenging data available at the moment. Seven semantic object classes are defined to label the LiDAR dataset and its corresponding street view images: building, tree, car, traffic sign, pedestrian, road, water (and

sky for unlabelled super-pixels in 2D images). Since the two other datasets do not contain street view images corresponding to LiDAR point clouds we use only this dataset to experiment 2D semantic segmentation.

*Paris-Rue-Madame* dataset presented in [34] is used to compare our method with other recent works on 3D segmentation and labelling. This dataset is used for urban detection-segmentation-classification methods, consists of accurate annotated 3D point clouds acquired by MLS system on Madame Street in Paris. The division of data to the training and test sets is described in [33,34] and we compare our results to their reported accuracies.

*TLS (terrestrial laser scanning) Velodyne* dataset [21] includes ten high-quality 3D point cloud scenes collected by a Velodyne LiDAR mounted on a car navigating through the Boston streets. Due to the specific nature of this dataset, we evaluate our method using each LiDAR rotation as a single scene (approximately 70,000 points).

*Performance measure* Both 3D LiDAR point segmentation and 2D street view segmentation are evaluated point/pixel-wise. We report accuracies for each label and compute other metrics, such as average precision, to compare to the existing works.

## 5.2 Urban 3D segmentation and classification

### Paris-Rue-Madame

The point-wise classification results for our method and for the two recently proposed methods by Aijazi et al. [1] and Serna and Marcotegui [33] are shown in Table 2. Our method achieved an average accuracy of 94.1% with notable margin of 8.5 and 22.2% with respect to existing methods [1] and [33], respectively. Note that even for this relatively easy dataset, the traffic sign class turned out to be particularly challenging due to lack of sufficient training samples. Significant performance deteriorations were observed for all methods:

**Table 2** Comparison of our method to other reported results on 3D point cloud classification with the Paris-Rue-Madame dataset

Method	Building	Road	Tr. sign	Car	Class AVE ACCY
Aijazi et al. [1]	0.914	0.901	0.710	0.900	0.856
Serna et al. [33]	0.986	0.940	0.000	0.950	0.719
Our	<b>0.991</b>	<b>0.950</b>	<b>0.841</b>	<b>0.982</b>	<b>0.941</b>

The bold numbers are related to best results comparing different methods

**Table 3** Comparison of our method to other reported results on 3D point cloud classification with the TLS Velodyne dataset

Measure	Method	Tree	Car	Tr. sign	Pedestrian	Fence	Building	Class AVE ACCY
Precision	Lai and Fox [21]	0.83	0.91	<b>0.80</b>	0.41	0.61	0.86	0.73
	Our	<b>0.89</b>	<b>0.95</b>	0.72	<b>0.88</b>	<b>0.85</b>	<b>0.95</b>	<b>0.87</b>
F-score	Lai and Fox [21]	0.76	0.79	<b>0.69</b>	0.47	0.42	0.91	0.67
	Our	<b>0.85</b>	<b>0.93</b>	<b>0.69</b>	<b>0.88</b>	<b>0.80</b>	<b>0.95</b>	<b>0.85</b>

The bold numbers are related to best results comparing different methods

the drop in our method was about 10% while for existing methods 15 and 71%, respectively.

### TLS Velodyne

It is notable that our algorithm was initially designed to analyse MLS LiDAR point clouds. One of the main advantages of our method is that it easily adapts to other types of LiDAR datasets such as terrestrial laser scanning (TLS) and airborne laser scanning (ALS) point clouds without major modification as long as the point units are in metric system (thresholds are set in metres). To exemplify this we evaluated our method with the same fixed parameters on the TLS Velodyne LiDAR dataset which contains 3D point clouds in local coordinate system of the LiDAR. The total number of points in each ten scene is nearly 70,000 and the average point density is about 12 points/m<sup>2</sup>. We compare our method to Lai and Fox [21]. We selected seven scenes for training and the three remaining scenes for testing similar to them and report per class average precision and F-score computed as

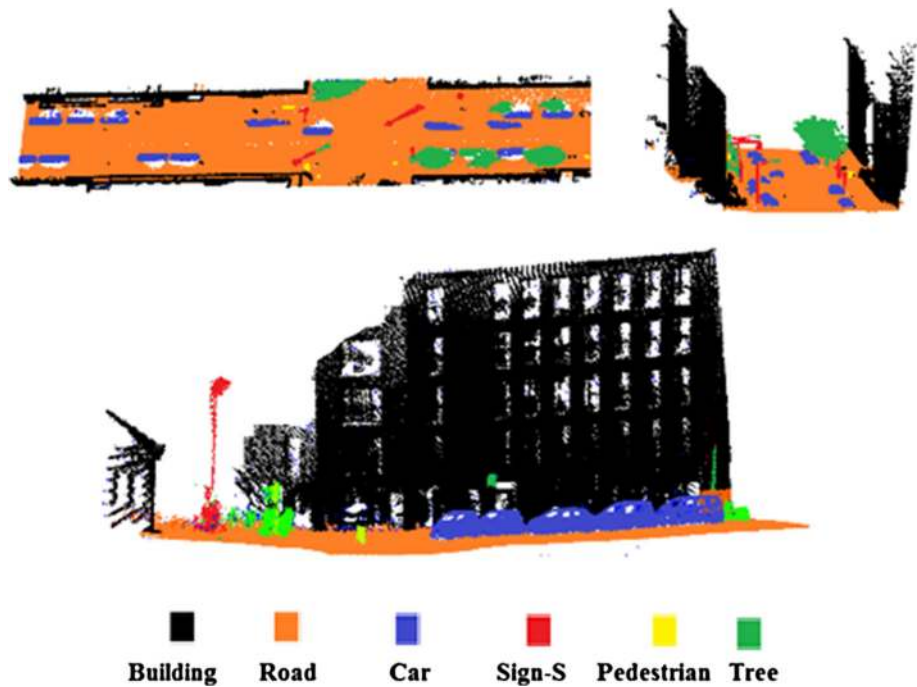
$$\frac{2 \times \text{recall} \times \text{precision}}{\text{recall} + \text{precision}}$$

The results in Table 3 show that for 5 out of 6 classes our method is clearly better and our F-score for each class is better than the average F-score of Lai and Fox.

### NAVTEQ True

The NAVTEQ True dataset is our main target - high-quality large-scale ground acquired dataset. NAVTEQ True collected from Boston, Paris and Chicago contains more than 80 million points and covers approximately 2.4 km of road altogether. Seven semantic object classes are defined to label the scenes: building, road, river, car, tree, traffic sign and pedestrian. The point clouds from the three cities are divided into two portions: the training set, and the testing set. The 70% of the total street length is selected for training and 30% for

**Fig. 9** Segmented and classified 3D LiDAR points of the NAVTEQ True dataset from Helsinki (colours encode the different labels)



**Table 4** Confusion matrix of our method for classification of the NAVTEQ True dataset

	Building	Road	River	Car	Tree	Tr. sign	Pedestrian
Building	0.885	0.083	0.000	0.000	0.115	0.000	0.000
Road	0.041	0.958	0.000	0.003	0.015	0.001	0.000
River	0.000	0.249	0.733	0.000	0.000	0.000	0.000
Car	0.000	0.018	0.000	0.847	0.000	0.007	0.000
Tree	0.004	0.001	0.000	0.000	0.897	0.007	0.000
Tr. sign	0.000	0.002	0.000	0.000	0.113	0.735	0.002
Pedestrian	0.000	0.049	0.000	0.000	0.008	0.000	0.782

**Table 5** Computing times of our method with and without the rule-based steps quick for road surface and building detection. Without the rule-based step all points are classified using the super-voxel and boosted decision tree method

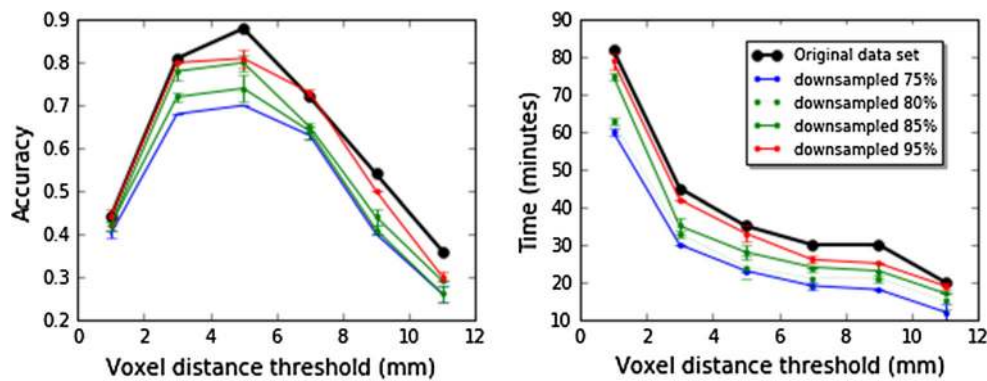
Our method	# of voxels	Comp. time (mins)	Overall accuracy
w rule-based	32,891	46	86%
w/o rule-based	246,548	291	75%

testing. Some typical results are illustrated in Fig. 9. Confusion matrix in Table 4 shows that the average accuracy (over all classes) is about 83%, with rule-based classification accuracies above 88%. Relatively low accuracies were reported for certain classes, e.g. pedestrian (78%), traffic sign (73%) and river (73%). These challenging cases are ascribed to the lack of sufficient training samples for each class.

#### Computing time

The proposed method has various advantages. The main contribution of this work is about achieving high accu-

racy within reasonable computing time. Considering the large-scale LiDAR datasets, we believe that fully supervised classification methods are computationally too expensive. In this experiment, we switched off the rule-based processing stage, but performed super-voxel-based supervised training and classification in Sect. 3.2. The results are collected to Table 5, and the computing time is wall time on Intel (R) Core(TM) i7-4710MQ 2.5 GHz CPU with 32 GB RAM. The results show that without the rule-based segmentation step the supervised classifier must construct and classify  $7.5\times$  more voxels and thus the computation time is  $6.3\times$  longer and requires much more memory usage. Moreover, without rule-based processing, the classification accuracy degraded significantly, partially due to the connectedness problem, i.e. roads surfaces and buildings are mis-segmented with other objects. In contrast, the removal of road surfaces and building facades created better isolated point clouds and hence improved classification accuracy from 75 to 86%.



**Fig. 10** Super-voxel classification accuracy on NAVTEQ True dataset (*left*) and computing time (*right*) with respect to the distance threshold  $\tau_{\text{voxel}}$  in (5)

**Table 6** Confusion matrix of pixel-wise accuracies of our method for 2D semantic classification of the NAVTEQ True street view images

	Sky	Building	Road	Tree	Car	Tr. sign	Pedestrian	River
Sky	0.960	0.020	0.000	0.020	0.000	0.000	0.000	0.000
Building	0.030	0.870	0.024	0.075	0.000	0.000	0.000	0.000
Road	0.000	0.015	0.920	0.000	0.065	0.000	0.000	0.000
Tree	0.000	0.280	0.080	0.640	0.000	0.000	0.000	0.000
Car	0.050	0.000	0.250	0.020	0.680	0.000	0.000	0.000
Tr. sign	0.010	0.280	0.090	0.000	0.000	0.370	0.250	0.000
Pedestrian	0.010	0.340	0.020	0.020	0.000	0.330	0.280	0.000
River	0.000	0.050	0.250	0.050	0.000	0.000	0.000	0.650

### Parameter settings

An important parameter controlling our method's accuracy and computing time is the threshold used to generate super-voxels (Sect. 3.2.1). In our experiments this was set to  $\tau_{\text{voxel}} = 0.005$  m, but to further study the effect of this parameter we conducted an ablation study with the NAVTEQ True dataset by varying the threshold value. The results of this experiment are shown in Fig. 10 (displayed in black curves) where the setting 5 mm clearly provides high accuracy with reasonable computation time.

The performance is evaluated in both robustness and accuracy terms with four sub-sampled of original point clouds. The testing point clouds are down-sampled uniformly to 75, 80, 85, 95% of the original point cloud density [31]. Refer Fig. 10 (colourful curves) for a summary of the algorithm performance results. Note that the dataset was down-sampled in multiple runs and the average accuracies as well as deviations were plotted in Fig. 10. The results show that the optimal threshold is consistent (around 5 mm) in despite that the average accuracy decreases with the percentage of down-sampling.

### 5.3 Semantic 2D segmentation

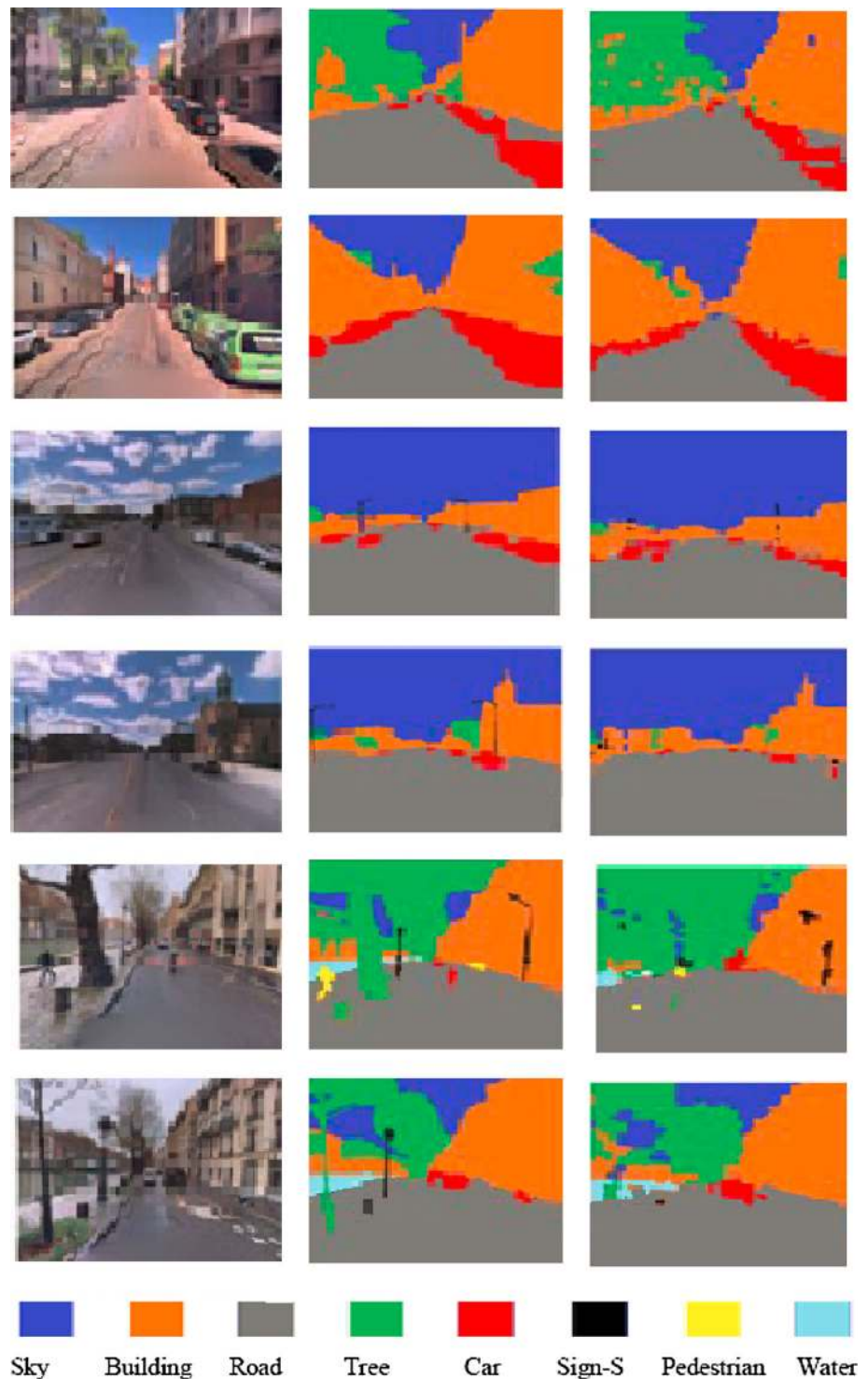
Dense scene labelling/segmentation is an important problem in robot and computer vision [24,29,44] and in our

case this can be achieved by backprojecting the labelled 3D points to 2D camera view plane (Sect. 4.2). For evaluation of 2D semantic segmentation we generated 2D ground truth by backprojecting the ground-truth 3D labels to the corresponding street view images in 500 randomly selected images in the NAVTEQ True test set. The back-projection results were manually verified and corrected. Pixel-wise classification accuracies are in Table 6. The sky, building and road regions were accurately labelled ( $\geq 85\%$  accuracy). The traffic signs and pedestrians were more poorly segmented, and this can be explained by the fact that there are not many examples for our classifier and therefore it makes misclassifications to the more frequent classes. However, the pixel-wise accuracies may give wrong interpretation of the results which qualitatively looked good as shown in the illustrative examples in Fig. 11.

## 6 Discussion

Firstly, rule-based classification is dedicated to the dominant objects, i.e. roads and buildings presented in LiDAR datasets, whereas rules are designed based on prior knowledge of these objects in terms of their sizes, relative positions, etc. A systematic approach to fine-tuning rules is to cross-validate rule parameters with respect to a separate dataset accompanied

**Fig. 11** 2D street view image segmentation using 3D label back projection. *Left* test image; *middle* ground truth; *right* our results



with ground-truth labels. Adding new rules can be treated in the similar manner. Nevertheless, a great deal of ground-truth labels is required to pursue this approach, making it only suitable for applications with ample ground-truth data available. Secondly, the street view 3D modelling application is restricted to the diversity and number of the pre-designed mesh templates in library. This problem can be solved by

creating a big library of street view objects such as trees and cars to generate more real 3D models.

### 7 Conclusions

We have proposed an efficient and accurate two-stage method to segment and semantically label urban 3D city maps of reg-

istered LiDAR point clouds and RGB street view images. Our method can process 80 million 3D points (2.4 km street distance) in less than an hour on commodity desktop hardware. The first processing stage uses rule-based detectors for road surfaces and building facades that span more than 75% of city point clouds. The rules are based on robust and adaptive processing (e.g. to the average building height of a specific city) with thresholds that have clear physical meaning and setting them is therefore intuitive. The remaining point cloud is processed by methodology that first constructs voxels (point clusters), and the super-voxels are then classified by an ensemble of boosted decision trees. Voxel construction, super-voxel construction and the extracted features are also based on thresholds and measures with clear physical meaning which allows their intuitive setting for other types of 3D map data. The rule-based stage makes computing  $6\times$  faster as compared to classifier-only and improves the segmentation accuracy. Moreover, we proposed two applications of our method: 1) model-based 3D visualization for better user experience and 2) 2D semantic segmentation for 2D applications. Both applications were also experimentally validated and our method performs favourably as compared to other existing methods. Our future work will address adaptation of the method for other 3D map data than urban city centres.

**Open Access** This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

## References

- Aijazi, A., Checchin, P., Trassoudaine, L.: Segmentation based classification of 3D urban point clouds: a super-voxel based approach with evaluation. *Remote Sens.* **5**(4), 1624–1650 (2013)
- Babahajiani, P., Fan, L., Kämäräinen, J.K., Gabbouj, M.: Automated super-voxel based features classification of urban environments by integrating 3D point cloud and image content. In: *IEEE International Conference on Signal and Image Processing Applications* (2015)
- Babahajiani, P., Fan, L., Kämäräinen, J.K., Gabbouj, M.: Comprehensive automated 3D urban environment modelling using terrestrial laser scanning point cloud. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshop on Large Scale 3D Data* (2016)
- Buch, A., Yang, Y., Krüger, N., Petersen, H.: In search of inliers: 3d correspondence by local and global voting. In: *CVPR* (2014)
- Collins, M., Schapire, R.E., Singer, Y.: Logistic regression, AdaBoost and Bregman distances. *Mach. Learn.* **48**(1), 253–285 (2002)
- Csurka, G., Dance, C., Fan, L., Willamowski, J., Bray, C.: Visual categorization with bags of keypoints. In: *ECCV Workshop* (2004)
- Dohan, D., Matejek, B., Funkhouser, T.: Learning hierarchical semantic segmentations of LIDAR data. In: *International Conference on 3D Vision (3DV)* (2015)
- Douillard, B., Underwood, J., Kuntz, N., Vlaskine, V., Quadros, A., Morton, P., Frenkel, A.: On the segmentation of 3d lidar point clouds. In: *ICRA* (2011)
- Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.* **32**(9), 1627–1645 (2010)
- Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: *CVPR* (2014)
- Golovinskiy, A., Kim, V., Funkhouser, T.: Shape-based recognition of 3d point clouds in urban environments. In: *ICCV* (2009)
- Guo, B., Huang, X., Zhang, F., Sohn, G.: Classification of airborne laser scanning data using JointBoost. *Isprs J. Photogramm. Remote sens.* **100**, 71–83 (2015)
- Guo, Y., Bennamoun, M., Sohel, F., Lu, M., Wan, J.: 3d object recognition in cluttered scenes with local surface features: a survey. *PAMI* **36**(11), 2270–2287 (2014)
- Guo, Y., Bennamoun, M., Sohel, F., Lu, M., Wan, J., Kwok, N.: A comprehensive performance evaluation of 3D local feature descriptors. *Int. J. Comput. Vis.* **116**, 66–89 (2016)
- Guo, Y., Sohel, F., Bennamoun, M., Lu, M., Wan, J.: Rotational projection statistics for 3d local surface description and object recognition. *Int. J. Comput. Vis.* **105**(1), 63–86 (2013)
- Hartley, R., Zisserman, A.: *Multiple View Geometry in Computer Vision*. Cambridge press, Cambridge (2003)
- Käler, O., Reid, I.: Efficient 3d scene labeling using fields of trees. In: *ICCV* (2013)
- Knopp, J., Prasad, M., Willems, G., Timofte, R., van Gool, L.: Hough transform and 3D SURF for robust three dimensional classification. In: *ECCV* (2010)
- Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: *NIPS* (2012)
- Lafarge, F., Mallet, C.: Creating large-scale city models from 3d-point clouds: a robust approach with hybrid representation. *Int. J. Comput. Vis.* **99**(1), 69–85 (2012)
- Lai, K., Fox, D.: Object recognition in 3d point clouds using web data and domain adaptation. *Int. J. Robot. Res.* **29**(8), 1019–1037 (2010)
- Lantuejoul, C., Maisonneuve, F.: Geodesic methods in quantitative image analysis. *Pattern Recognit.* **17**(2), 177–187 (1984)
- Li, J., Li, X., Yang, B., Sun, X.: Segmentation-based image copy-move forgery detection scheme. *IEEE Trans. Inf. Forensics Secur.* **10**(3), 507–518 (2015)
- Martinovic, A., Knopp, J., Riemenschneider, H., Van Gool, L.: 3D all the way: Semantic segmentation of urban scenes from start to end in 3D. In: *CVPR* (2015)
- Max, N.: Logistic regression, AdaBoost and Bregman distances. *Mach. Learn.* **48**(1–3), 253–285 (2002)
- Musialski, P., Wonka, P., Aliaga, D., Wimmer, M., van Gool, L., Purgathofer, W.: A survey of urban reconstruction. *Comput. Graph. Forum* **32**(6), 146–177 (2013). doi:10.1111/cgf.12077
- Nguyen, A., Le, B.: 3d point cloud segmentation: a survey. In: *IEEE Conference on Robotics, Automation and Mechatronics (RAM)* (2013)
- Papazov, C., Burschka, D.: An efficient RANSAC for 3D object recognition in noisy and occluded scenes. In: *ACCV* (2010)
- Ren, X., Bo, L., Fox, D.: RGB-(D) scene labeling: features and algorithms. In: *CVPR* (2012)
- Rusinkiewicz, S., Levoy, M.: Efficient variants of the ICP algorithm. In: *International Conference on 3D Digital Imaging and Video* (2001)
- Rusu, R.B., Cousins, S.: 3D is here: Point Cloud Library (PCL). In: *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai (2011)
- Sengupta, S., Greveson, E., Shahrokni, A., Torr, P.: Urban 3d semantic modelling using stereo vision. In: *ICRA* (2013)

33. Serna, A., Marcotegui, B.: Detection, segmentation and classification of 3D urban objects using mathematical morphology and supervised learning. *ISPRS J. Photogramm. Remote Sens.* **93**, 243–255 (2014)
34. Serna, A., Marcotegui, B., Goulette, F., Deschaud, J.E.: Paris-rue-Madame database: a 3D mobile laser scanner dataset for benchmarking urban detection, segmentation and classification methods. In: 3rd International Conference on Pattern Recognition, Applications and Methods ICPRAM 2014 (2014)
35. Sivic, J., Zisserman, A.: Video google: A text retrieval approach to object matching in videos. In: *ICCV* (2003)
36. Tarini, M., Cignoni, P., Scopigno, R.: Visibility based methods and assessment for detail-recovery. In: *Proceedings of IEEE Visualization*, pp. 60–67 (2003)
37. Theologou, P., Pratikakis, I., Theoharis, T.: A comprehensive overview of methodologies and performance evaluation frameworks in 3d mesh segmentation. *Comput. Vis. Image Underst.* **135**, 49–82 (2015)
38. Velizhev, A., Shapovalov, R., Schindler, K.: Implicit shape models for object detection in 3d point clouds. In: *ISPRS Congress* (2012)
39. Xiao, J., Quan, L.: Multiple view semantic segmentation for street view images. In: *ICCV* (2009)
40. Xie, J., Kiefel, M., Sun, M., Geiger, A.: Semantic instance annotation of street scenes by 3d to 2d label transfer. In: *CVPR* (2016)
41. Xu, C., Whitt, S., Corso, J. J.: Flattening supervoxel hierarchies by the uniform entropy slice. In: *CVPR* (2013)
42. Yu, T., Wang, R.: Scene parsing using graph matching on street-view data. *Comput. Vis. Image Underst.* **145**, 70–80 (2016)
43. Zaharescu, A., Boyer, E., Horaud, R.: Keypoints and local descriptors of scalar functions on 2d manifolds. *Int. J. Comput. Vis.* **100**, 78–98 (2012)
44. Zhang, R., Candra, S., Vetter, K., Zakhor, A.: Sensor fusion for semantic segmentation of urban scenes. In: *CVPR* (2015)
45. Zheng, Y., Jeon, B., Xu, D., Wu, Q., Zhang, H.: Image segmentation by generalized hierarchical fuzzy c-means algorithm. *J. Intell. Fuzzy Syst.* **28**(2), 961–973 (2015)
46. Zhou, Z., Wang, Y., Wu, Q.J., Yang, C.N., Sun, X.: Effective and efficient global context verification for image copy detection. *IEEE Trans. Inf. Forensics Secur.* **12**(1), 48–63 (2017)



**Pouria Babahajiani** received his BS degree in Biomedical Engineering, Signal Processing, from Tehran Polytechnic (a.k.a Amirkabir University of Tech), and his MS in Electrical Engineering, Signal Processing from Tampere University of Technology, Tampere, Finland. Currently, he is continuing research in the field of Computer Vision as a PhD candidate in Tampere University of Technology and as an external researcher in Nokia Technologies. His research interests

include big data analytics, augmented and virtual reality, multimedia, convolutional neural networks, nonlinear signal and image processing, and 3D data processing and rendering.



**Dr Lixin Fan** is a principal scientist at Nokia Technologies. His research areas of interests include 3D image and video processing, computer vision, machine learning, big data analysis, intelligent human-computer interface, augmented and virtual reality, mobile ubiquitous and pervasive computing. His recent research activities are related to Nokia/HERE map 3D data processing and rendering. Dr Fan is the (co)author of more than 40 international journal & conference publications. He also (co)invented dozens of granted and pending patents filed in USA, Europe and China. Before joining Nokia in 2004, Dr Fan was affiliated with Xerox Research Center Europe and his research work included the well recognized Bag of Keypoints method for image categorization. Dr Fan received his MSc and PhD in Computer Science from National University of Singapore in 1998 and 2002, respectively.



**Dr. Joni-Kristian Kämäräinen** is associate professor of signal processing at the Department of Signal Processing, Tampere University of Technology, Finland. He holds MSc and PhD degrees from Lappeenranta University of Technology in 1999 and 2003, respectively. He leads the Computer Vision Group and his research focuses on 2D and 3D scene analysis, object detection and recognition, signal processing and machine intelligence.



**Moncef Gabbouj** received his BS degree in electrical engineering in 1985 from Oklahoma State University, Stillwater, and his MS and PhD degrees in electrical engineering from Purdue University, West Lafayette, Indiana, in 1986 and 1989, respectively. Dr. Gabbouj is a Professor of Signal Processing at the Department of Signal Processing, Tampere University of Technology, Tampere, Finland. He was Academy of Finland Professor during 2011–2015. He held several visiting professorships at different universities. Dr. Gabbouj is currently the TUT-Site Director of the NSF I/UCRC funded Center for Visual and Decision Informatics. His research interests include Big Data analytics, multimedia content-based analysis, indexing and retrieval, machine

learning, pattern recognition, nonlinear signal and image processing and analysis, voice conversion, and video processing and coding. Dr. Gabbouj is a Fellow of the IEEE and member of the Academia Europaea and the Finnish Academy of Science and Letters. He is the past Chairman of the IEEE CAS TC on DSP and committee member of the IEEE Fourier Award for Signal Processing. He served as Distinguished Lecturer for

the IEEE CASS. He served as associate editor and guest editor of many IEEE, and international journals. Dr. Gabbouj was the recipient of the 2015 TUT Foundation Grand Award, the 2012 Nokia Foundation Visiting Professor Award, the 2005 Nokia Foundation Recognition Award, and several Best Paper Awards. He published over 650 publications and supervised 45 doctoral and 58 Master theses.