A Model-based Approach for Fast Vehicle Detection in Continuously Streamed Urban LIDAR Point Clouds

Attila Börcs, Balázs Nagy, Milán Baticz and Csaba Benedek*

Distributed Events Analysis Research Laboratory,
Institute for Computer Science and Control of the Hungarian Academy of Sciences
H-1111, Kende utca 13-17 Budapest, Hungary
E-mail: firstname.lastname@sztaki.mta.hu

Abstract. Detection of vehicles in crowded 3-D urban scenes is a challenging problem in many computer vision related research fields, such as robot perception, autonomous driving, self-localization, and mapping. In this paper we present a model-based approach to solve the recognition problem from 3-D range data. In particular, we aim to detect and recognize vehicles from continuously streamed LIDAR point cloud sequences of a rotating multi-beam laser scanner. The end-to-end pipeline of our framework working on the raw streams of 3-D urban laser data consists of three steps 1) producing distinct groups of points which represent different urban objects 2) extracting reliable 3-D shape descriptors specifically designed for vehicles, considering the need for fast processing speed 3) executing binary classification on the extracted descriptors in order to perform vehicle detection. The extraction of our efficient shape descriptors provides a significant speedup with and increased detection accuracy compared to a PCA based 3-D bounding box fitting method used as baseline.

1 Introduction

1.1 Problem Statement

Efficient and fast perception of the surrounding environment has a major impact in mobile robotics research with many prominent application areas, such as autonomous driving, driving assistance systems, self localization and mapping, and obstacle avoidance [1, 2]. Future mobile vision systems promise a number of benefits for the society, including prevention of road accidents by constantly monitoring the surrounding vehicles or ensuring more comfort and convenience for the drivers. Outdoor laser scanners, such as LIDAR mapping systems particularly have become an important tools for gathering data flow for these tasks

* This work was partially funded by the Government of Hungary through a European Space Agency (ESA) Contract under the Plan for European Cooperating States (PECS), and by the Hungarian Research Fund (OTKA #101598).

since they are able to rapidly acquire large-scale 3-D point cloud data for real-time vision, with jointly providing accurate 3-D geometrical information of the scene, and additional features about the reflection properties and compactness of the surfaces. Moreover, LIDAR sensors have a number of benefits in contrast to conventional camera systems, *e.g.* they are highly robust against daily illumination changes, and they may provide a larger field of view. Robust detection and recognition of vehicles in 3-D urban scenarios is one of the major challenges in any robot perception related task. In this paper we focus on the vehicle detection problem relying on large-scale terrestrial point clouds recorded in different crowded urban scenarios, such as main roads, narrows streets and wide intersections. More specifically we use as input a raw point cloud stream of a rotating multi-beam (RMB) laser acquisition system. The problem of detection and recognition of certain types of object characteristics on streaming point clouds is challenging for various reasons. First, the raw measurements are noisy and contain several different objects in cluttered regions. Second, in crowded scenes the vehicles, pedestrians, trees and street furnitures often occlude each other causing missing or broken object parts in the visible measurement streams. Third, typically by terrestrial laser scanning the point cloud density rapidly decreases as a function of the distance from the sensor [3], which fact may cause strongly corrupted geometric properties of the object appearances, misleading the recognition modules. Further requirements arise for navigation or autonomous driving systems, where the data is continuously streamed from a laser sensor mounted onto a moving platform, and we are forced to complete the object detection and recognition tasks within a very limited time frame.

1.2 Related Works

Significant research efforts are expended nowadays for solving object recognition problems in point clouds obtained by 3-D laser scanners. Extracting efficient object descriptors (i.e. *features*) is an essential part in each existing technique, which step usually implements one of the two following strategies.

According to the *first* strategy, the shape and the size of the objects are approximated by 3-D bounding boxes. In [4] a framework has been proposed for object classification and tracking. The basic idea is to use an octree based Occupancy Grid representation to model the surrounding environment, and simple features for object classification, such as the length ratios of object bounding boxes. In that method three different object classes are considered: pedestrians, bicycles and vehicles. In our case, however, the observed environment consists of complex urban scenarios with many object types such as trees, poles, traffic signs, and occluded wall regions. Here simple features may not be robust enough for efficient object classification, due to the largely diverse appearances of the considered object shapes throughout an entire city. Other approaches derive 3-D bounding boxes for recognition via Principal Component Analysis (PCA) techniques. The authors of [5] and [6] calculate statistical point cloud descriptors: they compute saliency features which capture the spatial distribution of points in a local neighborhood by covariance analysis. The main orientation of

an object is derived from the principal components (eigenvalues and corresponding eigenvectors), considering the covariance matrix of the 3-D point positions. Object classification is achieved by three saliency features, namely *scatter*ness, *linear*ness and *surface*ness, which are calculated as linear combinations of the eigenvalues.

Following the *second* strategy, a group of existing object classification techniques use different features, based on shape and contextual descriptors [7], [8], [9]. [7] propose a system for object recognition, by clustering nearby points from a set of potential object locations. Thereafter, they assign the points near the estimated object locations to foreground and background sets using a graph-cut algorithm. Finally a feature vector is built for each point cluster, and the feature vectors are labeled by a classifier, trained on a manually collected object set. In [8] an algorithm is presented for fast segmentation of point cloud regions, followed by a 3-D segment classification step which is based on various 3-D features, like the spin image or the spherical harmonic descriptor. [9] introduce an approach for detecting and classifying different urban objects from a raw stream of 3-D laser data such as cars, pedestrians and bicyclists. For this purpose a graph-based clustering algorithm, different shape descriptors and shape functions are exploited, such as the spin image and PCA based eigenvectors. Although these general shape and context based methods may provide more precise recognition rates for urban objects than the 3-D bounding box based techniques, they are computationally more expensive, and often do not perform in real time.

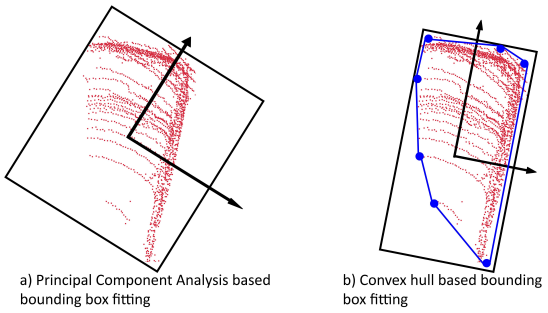

Fig. 1. Demonstrating the limitations of PCA based bounding box approximation, and the advantages of the proposed convex hull based bounding box fitting technique on the top-view projection of a selected vehicle in the point cloud

2 Contributions of the Proposed Approach

In this work we present a real-time model-based system for vehicle detection and extraction from continuously streamed LIDAR point clouds, which are captured in challenging urban scenarios. By constructing the proposed vehicle model, we

combine three novel feature extraction steps. *First* we propose a new convex hull based 2-D bounding box fitting method, which is used for fast and precise estimation of the location, size and orientation parameters of the vehicle candidates. *Second*, we develop a 3-D sphere based feature, which is used for approximating the principal curvatures of the objects in 3-D. *Third*, we extract the object contours from the side-view, in order to obtain a representative shape characteristics of vehicles in 2-D. Our model gives two major contributions over existing approaches:

◇ *Fast 2-D bounding box fitting for cluttered and partially incomplete objects:* It is highly challenging to fit precise bounding boxes around the objects in RMB LIDAR range data streams, since we should expect various artifacts of self-occlusion, occlusion by other object, measurement noise, inhomogeneous point density and mirroring effects. These factors drastically change the appearances of the 3-D objects, and the conventional principal component analysis (PCA) based techniques [4, 5] may not give sufficient results. Especially, in the RMB point cloud streams, only the object side facing the sensor is clearly visible, and the opposite side of the object is usually completely or partially missing. For this reason, if we calculate by PCA covariance analysis the principal directions of a point cloud segment identified as a vehicle candidate, the eigenvectors usually do not point towards the main axes of the object, yielding inaccurately oriented bounding boxes, as demonstrated in Fig. 1. In contrast to PCA solutions in 3-D, we calculate the 2-D convex hull of the top-view projection of the objects, and we derive the 2-D bounding boxes directly from the convex hull. As shown later, this strategy is less sensitive to the inhomogeneous point density and the presence of missing/occluded object segments, since instead of calculating spatial point distributions for the entire object's point set, we capture here the local shape characteristics of the visible object parts, and fit appropriate 2-D bounding boxes with partial matching.

◇ *Lightweight shape analysis for streaming data:* For enhancing the classification performance of object recognition, an efficient shape descriptor, called the spin image, has been adopted in several previous methods [7–9]. Spin image based features can be used to approximate the object shapes by surface meshes, yielding robust solutions for object classification and recognition. However, the demand of real-time performance is not feasible here, since estimating different surface models for 3-D data is a computationally expensive task. Therefore, these mesh based models are not directly designed for continuously streamed range data. In our solution, we propose two new features for approximating the principal curvatures of the objects in 3-D. The extracted contour of the detected object's side-view profile can be compared to a reference vehicle contour model, which is obtained by supervised training. To train our classifier, we use vehicle samples from a manually annotated point cloud database. Our proposed features are also able to sufficiently model the shape characteristics of vehicle objects, while they can

be calculated very quickly, offering a decent trade-off with respect to speed and accuracy.

The description of the proposed model-based recognition framework is structured as follows. In Section 3 we briefly present the preprocessing steps. First, a hierarchical grid based data structure is introduced, which will allow us to perform fast retrieval of 3-D point cloud features for segmentation and detection purposes. Second, we propose a point cloud segmentation algorithm, to distinguish the *foreground* regions of the captured scene containing the moving or static field objects, from the *background* composed of the roads and other terrain parts. Third, a fast connected component analysis algorithm is presented for separating individual objects within the foreground regions. The main new contributions of this paper are related to vehicle detection and localization. In Section 4, we introduce our *vehicle model* with defining an efficient set of features, which characterize vehicle candidates in the point clouds. In Section 5, a Support Vector Machine (SVM) based object classification process is described based on the previous features. Finally we report on the experiments and present the evaluation results in Section 6.

3 Point Cloud Segmentation and Object Separation

In this section, we introduce the point cloud preprocessing module of the proposed system to prepare the data for the vehicle detection step. An efficient grid based method was presented [10, 11], which will be adopted here for robust foreground extraction and 3-D object separation, in challenging dense urban environments where several nearby object may be located close to each other.

◇ *The Hierarchical Grid Model data structure:*

We fit a regular 2-D grid S with W_S rectangle side length onto the $P_{z=0}$ plane (using the RMB sensor's vertical axis as the z direction and the sensor height as a reference coordinate), where $s \in S$ denotes a single cell. We assign each $p \in \mathcal{P}$ point of the point cloud to the corresponding cell s_p, which contains the projection of p to $P_{z=0}$. Let us denote by $\mathcal{P}_s = \{p \in \mathcal{P} : s = s_p\}$ the point set projected to cell s. Moreover, we store the height coordinate and different height properties such as, maximum $z_{\max}(s)$, minimum $z_{\min}(s)$ and average $\hat{z}(s)$ of the elevation values within cell s, which quantities will be used later for foreground separation.

For robust object detection a denser grid resolution is also required in this grid data structure, therefore the cell s of the coarse grid level is subdivided into smaller cells $s'_d | d \in \{1, 2, \ldots, \xi^2\}$, with cell side length $W_{s'_d} = W_s/\xi$, where ξ is a scaling factor (used $\xi = 3$). We store references for each 3-D point in the coarse and dense grid levels as well.

◇ *Foreground separation and object detection:*

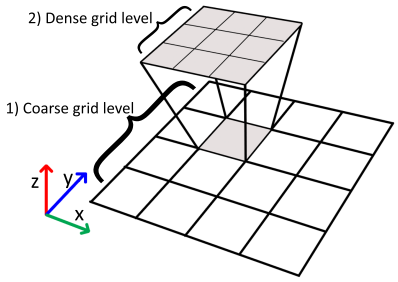

Fig. 2. Visualization of our *hierarchical grid model* data structure - *(bottom)* the coarse grid level: the 3-D space coarsely quantized into 2-D grid cells, *(top)* the dense grid level: each grid cell on the coarse level subdivided into smaller cells.

The foreground separation is achieved on the coarse grid level of the above presented grid data structure. Our goal is to discriminate foreground regions *e.g.* street furnitures, pedestrians, vehicles, walls and other street objects, and background regions composed of the roads, sidewalks and grassy ground. For ground segmentation we apply a locally adaptive terrain modeling approach similarly to [10], which is able to accurately extract the road regions, even if their surfaces are not perfectly planar. We use point height information for assigning each grid cell to an appropriate cell class. Before that, we detect and remove grid cells that belong to irrelevant *clutter* regions, thus we will not visit these cells later and save processing time. We classify each cell to *clutter*, which contains less points than a predefined threshold (typically 4-8 points). After clutter removal all the points in a cell are classified as *ground*, if the difference of the minimal and maximal point elevations in the cell is smaller than a threshold (used 25cm), moreover the average of the elevations in neighboring cells does not exceeds an allowed height range based on a globally estimated digital terrain map. The first criterion ensures the flatness or homogeneity of the points. Given a cell with 60 centimeters of width, this allows 22.6° of elevation within a cell; higher elevations are rarely expected in an urban scene. The rest of the points in the cloud are assigned to class *foreground* belonging to vehicles, pedestrians, mail boxes, billboards etc.

After the foreground separation step, our aim is to find distinct groups of points which belong to different urban objects on the foreground. For this task we use the *hierarchical grid model*: On one hand, the coarse grid resolution is appropriate for a rough estimation of the 3-D blobs in the scene, in this way we can also roughly estimate the size and the location of possible object candidates. On the other hand, using a dense grid resolution beside a coarse grid level, is efficient for calculate point cloud features from a smaller subvolume of space, therefore we can refine the detection result derived from the coarse grid resolution.

The proposed object detection algorithm consists of three main steps: *First*, we visit every cell of the coarse grid and for each cell s we consider the cells

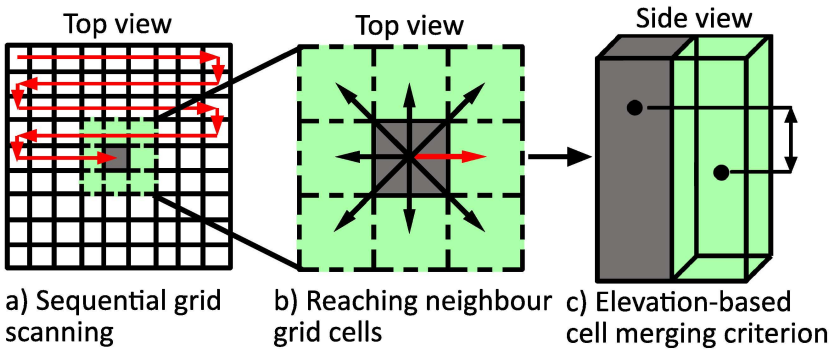

Fig. 3. The step by step demonstration of the object detection algorithm

in its 3×3 neighborhood (see Fig. 3a),3b)). We visit the neighbor cells one after the other in order to calculate two different point cloud features: (i) the maximal elevation value $Z_{max}(s)$ within a coarse grid cell and (ii) the point cloud density (*i.e.* point cardinality) of a dense grid cell. *Second* our intention is to find connected 3-D blobs within the foreground regions, by merging the coarse level grid cells together. We use and elevation-based cell merging criterion for perform this step. $\psi(s, s_r) = |Z_{max}(s) - Z_{max}(s_r)|$ is a merging indicator, which measures the difference between the maximal point elevation within cell s and its neighboring cell s_r. If the ψ indicator is smaller than a predefined value, we assume that s and s_r belong to the same 3-D object (see Fig. 3c)). *Third*, we perform a detection refinement step on the dense grid level. The elevation based cell merging criterion on the coarse grid level often yields that nearby and self-occluded objects are merged into a same blob. We handle this issue by measuring the point density in each sub-cell s'_d at the dense grid level. Our assumption is here that the nearby objects, which were erroneously merged at the coarse level, could be appropriately separated at the fine level, as the examples in Fig. 4 show. Let us present three typical urban scenarios when the *simple* coarse grid model merges the close objects to the same extracted component, while using a *hierarchical* grid model with coarse and dense grid level, the objects can be appropriately separated. We consider two neighboring super-cell pairs -marked by red - in Fig. 4a) and Fig. 4b), respectively. In both cases the cells contain points from different objects, which fact cannot be justified at the coarse cell level. However, at the dense level, we can identify connected regions of near-empty sub-cells (denoted by gray), which separate the two objects. Fig. 4c) demonstrates a third configuration, when a super-cell intersects with two objects, but at the sub-cell level, we can even find a separator line.

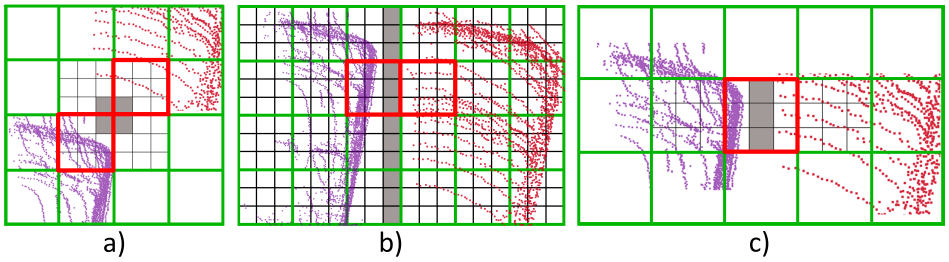

Fig. 4. Separation of close objects at the dense grid level. [color codes: green lines =coarse grid level, black lines=dense grid level, grey cells= examined regions for object separation]

4 Vehicle Model and Feature Extraction

The output of the workflow introduced in Section 3 is a list of point cloud segments (called hereafter *blobs*) representing the object candidates of the scene. Our next goal is to identify the vehicles among the extracted blobs.

In this section we present our features used for point cloud based *vehicle modeling*, and the feature integration process. The proposed module consists of the combination of three descriptors in order the find the optimal trade-off w.r.t. speed and accuracy. First, we approximate the position, size, and orientation parameters of the obtained objects by fitting a 2-D bounding box to the point cloud blobs. In contrast to previous works [4–6] we do not calculate spatial point distributions over the entire blobs, due to reasons detailed in Sec. 2. Instead, we determine the vehicle's 2-D convex hull with a matching step considering only the visible structure elements. In this way our solution is independent of the local point cloud density at the given object position, and performs efficiently even if a significant part of the object is occluded. On the other hand, we have observed that the special curvature of the vehicle shape, especially around the windshields, is a characteristic feature for visual recognition, which can be quickly estimated and efficiently used for identification. The detailed explanation of the proposed feature extraction strategy is presented as follows:

◇ *2-D bounding box fitting via convex hulls:* Let us consider the output of the preprocessing module (Section 3) on the dense grid level of the *Hierarchical Grid Model*. At this step, we only use the width and a depth (X,Y) coordinates of the points, and the height coordinates (Z) are ignored. We mark *first* which cells are occupied by several points, and which ones are empty. Next we visit the 3×3 neighborhood of each occupied cell, and filter out the cells, where all neighbors are occupied as well. In this way we can roughly estimate the boundary cells of the object. Then, we construct the convex hull from the points of the boundary cells using the monotone chain algorithm [12]. In the forthcoming key step, we attempt to fit the *optimal* 2-D bounding box to the convex hull as follows (see also a demonstration in Fig. 5):

- Visit the consecutive point pairs of the hull p_i and p_{i+1}, one after another ($i = 1, 2, \ldots, i_{\max}$):
  1. Consider the line l_i between point p_i and p_{i+1}, as a side candidate of the bounding box rectangle.
  2. Find the $p_\star$ point of the hull, whose distance is maximal from l_i, and draw a $l_\star$ parallel line with l_i which intersects $p_\star$. We consider $l_\star$ as the second side candidate of the bounding box.
  3. Project all the points of the convex hull to the line l_i, and find the two extreme ones p' and p''. The remaining two sides of the bounding box candidate will be constructed by taking perpendicular lines to l_i, which intersect p' and p'' respectively.
- Chose the optimal bounding box from the above generated rectangle set by minimizing the average distance between the points of the convex hull and the fitted rectangle.

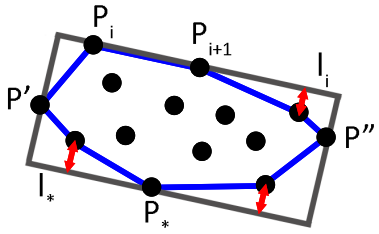

Fig. 5. Demonstration of the fast 2-D bounding box fitting algorithm for the convex hull of the top-view object projection (the bounding box is shown marked by gray color

◇ *Principal curvature estimation with 3-D spheres:* Our aim here is to place four spheres near to the four top corners of the vehicle's roof, in order to examine the typical curvatures around this regions, especially close to the windshields. First we vertically shift the 2-D bounding box obtained by the previous feature extraction step to the maximal elevation within the vehicle's point set. This configuration is demonstrated in Fig. 6 both from top-view and from side-view. Then we set the center points of the spheres to the corner points of the shifted 2-D bounding rectangle. We start to increase the radius of the spheres as long as they hit a 3-D point from the vehicle. Our assumption is that due to the typical slope around the windshields, we should experience significant differences between the radiuses of the four spheres. We can also observe in Fig. 6 that the radiuses of the frontal spheres (R_1 and R_2) are significantly larger than the radiuses of the spheres at the back side (R_3 and R_4). We use in the following the four radius values in the object's feature vector.

◇ *Shape approximation of side-view profile using convex and concave hulls:* at this step, we project the point clouds of the object candidates to a verti-

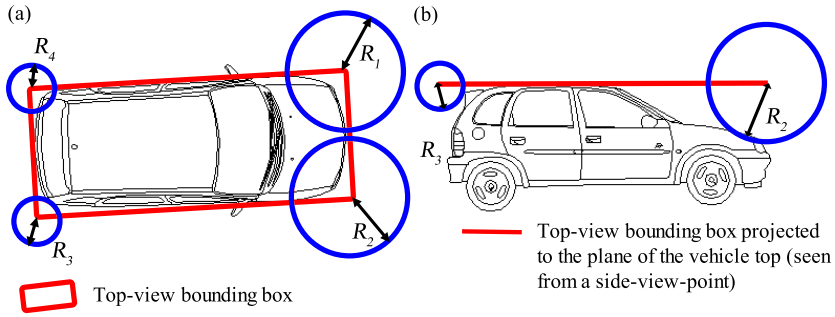

Fig. 6. Demonstration of the principal curvature feature with 3-D spheres

cal plane which is parallel to the main axis of the top-view bounding box. Thereafter, we fit to the 2-D side-view object silhouettes a convex hull, and a concave hull with 20cm resolution. Here the shape features are the contour vectors of the convex and concave hulls themselves, so that we store the contours of sample vehicles with various prototypes in a library, and we compare the contours of the detected objects to the library objects via the turning function based polygon representation [13].

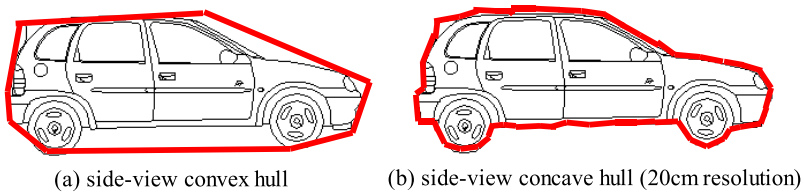

Fig. 7. Demonstration of side-view convex and concave hulls

5 Classification

In the final stage of the detection process, we must label each object candidate as *vehicle* or *background* (*i.e.* non-vehicle), based on a feature vector composed of the descriptors from Section 4. We used the following three feature components: 1) The length and the width of the approximated 2-D bounding box derived from the convex hull. 2) The four radius values of the 3-D spheres, as well as the radius difference between the frontal and the back sphere pairs, which are proposed for principal curvature estimation. 3) The difference between the concave side profile hull of the vehicle candidate and the prototype shape, which is a real

number normalized between zero and one. Consequently, the resulting feature vector consists of eight dimensions. Following a supervised approach, we created first a training set of positive and negative vehicle samples. For this purpose, we have developed a 3-D annotation tool, which enables labeling the urban objects in the point clouds as vehicles or background. We have manually collected more than 1600 positive samples (*i.e.* vehicles), and also generated 4000 negative samples from different scenarios recorded in the streets of Budapest, Hungary. The negative samples were created by a semi-automatic process, cutting random regions from the point clouds, which were manually verified. In addition, 12715 positive vehicle samples, and 3396 negative samples (different street furniture and other urban objects) have been selected from the KITTI Vision Benchmark Suite [14] and used for the training of the classifier. We have performed a binary classification between the *vehicles* and the *background objects* by a Support Vector Machine (SVM) using the toolkit from [15]. Here we used $C = 0.1078$ as the kernel function parameter, and $\nu = 0.0377$ as the upper bound on the fraction of margin errors.

6 Experiments

Point Cloud Dataset	NO	PCA based approach [5]		Prop. Model-based approach	
		F-rate(%)	Avg. Processing Speed (fps)	F-rate(%)	Avg. Processing Speed (fps)
Budapest Dataset #1	567	73	15	89	24
Budapest Dataset #2	1141	71	12	90	21
Budapest Dataset #3	368	57	13	80	22
KITTI Dataset [14]	614	62	14	78	25
Overall	2690	68	13.5	86	23

Table 1. Numerical comparison of the detection results obtained by the Principal Component Analysis based technique [5] and the proposed *Model-based framework*. The number of objects (NO) are listed for each data set, and also in aggregate.

We evaluated our method on four LIDAR point cloud sequences, concerning different types of urban scenarios, such as main roads, narrow streets and intersections. Three scenarios have been recorded in the streets of Budapest, Hungary, and the fourth scenario has been selected from the KITTI Vision Benchmark Suite [14]. All the test sequences have been recorded by a Velodyne HDL-64E S2 rotating multi-beam LIDAR sensor, with a 10Hz rotation speed. We have compared our *Model-based approach* to a reference method, which uses a simple occupancy grid representation for foreground separation, and applies Principal Component Analysis (PCA) based features for object classification [5].

Qualitative results of our proposed model on four sample frames are shown in Fig. 8.[1] During the quantitative evaluation, we verified the proposed method and the reference PCA based technique on 2690 vehicles, using the Ground Truth (GT) information. To enable fully automated evaluation, we needed to make first a non-ambiguous assignment between the detected objects and GT object samples, where we used the Hungarian algorithm [16] to find optimal matching. Thereafter, we counted the number of Missing Objects (MO), and the Falsely detected Objects (FO). These values were compared to the Number of real Objects (NO), and the F-rate of the detection (harmonic mean of precision and recall) was also calculated. We have also measured the processing speed of the two methods in frames per seconds (fps). The numerical performance analysis is given in Table 1. The processing speed of the individual modules of the framework on the entire test dataset, which consist of 2690 vehicles is the following: 1) Building hierarchical grid data structure - *13ms* 2) Point cloud segmentation - *4ms* 3) Object separation step - *6ms* 4) Feature extraction step - *18ms* 5) SVM decision *2ms*. The results confirm that the proposed model surpasses the PCA based method in F-rate for all the scenes. Moreover, the proposed *Model-based approach* is significantly faster on the streaming data, and in particularly, it gives more reliable results in the challenging crowded urban scenarios (#2 and #3), where several vehicles are occluded by each other, and the scene contains various types of other objects and street furnitures such as walls, traffic signs, billboards, pedestrians etc. The proposed method only fails on highly occluded vehicles, where the objects are broken into many parts or the majority of the vehicle's point cloud is missing. As for the computational speed, we measured 13.5 fps in average with the Principal Component Analysis based technique [5] and 23 fps with the proposed *Model-based approach*.

7 Conclusion

In this work we have proposed a novel *Model-based framework*, which uses three new descriptors for robust detection of vehicles in continuously streamed point cloud sequences of a rotating multi-beam LIDAR sensor. Due to the presented features we have observed a reliable performance in challenging dense urban scenarios with multiple occlusions and the presence of various types of scene objects. The model has been quantitatively validated based on Ground Truth data, and the advantages of the proposed solution versus a state-of-the-art technique have been demonstrated.

References

1. McNaughton, M., Urmson, C., Dolan, J.M., Lee, J.W.: Motion planning for autonomous driving with a conformal spatiotemporal lattice. In: ICRA. (2011) 4889–4895

[1] Demonstration videos are also available at the following url: https://vimeo.com/pointcloudprocessing

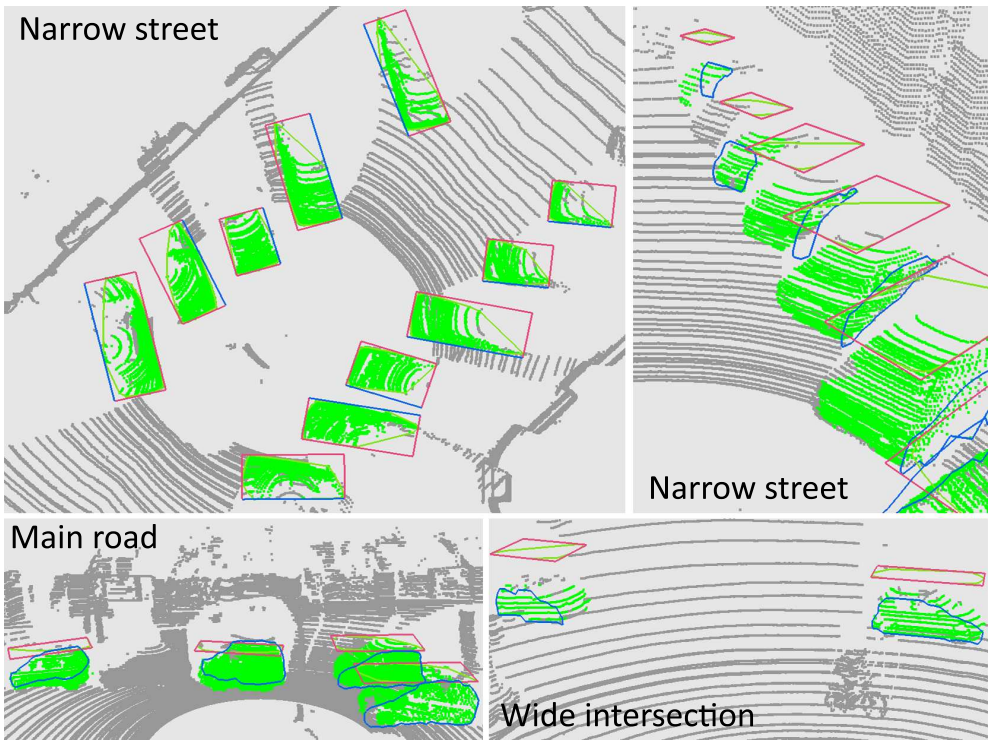

Fig. 8. Qualitative results of vehicle detection with displaying the top-view bounding boxes (by red) and the side view concave hulls (blue) extracted by the algorithm

2. Levinson, J., Montemerlo, M., Thrun, S.: Map-based precision vehicle localization in urban environments. In: Proceedings of Robotics: Science and Systems, Atlanta, GA, USA (2007)
3. Behley, J., Steinhage, V., Cremers, A.B.: Performance of histogram descriptors for the classification of 3d laser range data in urban environments. In: ICRA, (IEEE) 4391–4398
4. Azim, A., Aycard, O.: Detection, classification and tracking of moving objects in a 3D environment. In: IEEE Intelligent Vehicles Symposium (IV), Alcalá de Henares, Spain (2012) 802–807
5. Himmelsbach, M., Müller, A., Luettel, T., Wuensche, H.J.: LIDAR-based 3D Object Perception. In: Proceedings of 1st International Workshop on Cognition for Technical Systems, Munich (2008)
6. Lalonde, J.F., Vandapel, N., Huber, D., Hebert, M.: Natural terrain classification using three-dimensional ladar data for ground robot mobility. Journal of Field Robotics **23** (2006) 839 – 861
7. Golovinskiy, A., Kim, V.G., Funkhouser, T.: Shape-based recognition of 3D point clouds in urban environments, Kyoto, Japan (2009)
8. Douillard, B., Underwood, J., Vlaskine, V., Quadros, A., Singh, S.: A pipeline for the segmentation and classification of 3d point clouds. In: In ISER. (2010)
9. Wang, D.Z., Posner, I., Newman, P.: What could move? finding cars, pedestrians and bicyclists in 3d laser data. In: Proc. IEEE International Conference on Robotics and Automation (ICRA), Minnesota, USA (2012)
10. Józsa, O., Börcs, A., Benedek, C.: Towards 4D virtual city reconstruction from Lidar point cloud sequences. In: ISPRS Workshop on 3D Virtual City Modeling. Volume II-3/W1 of ISPRS Annals Photogram. Rem. Sens. and Spat. Inf. Sci., Regina, Canada (2013) 15–20
11. Börcs, A., Nagy, B., Benedek, C.: Fast 3-D urban object detection on streaming point clouds. In: Workshop on Computer Vision for Road Scene Understanding and Autonomous Driving at ECCV, Lecture Notes in Computer Science, Zürich, Switzerland (2014)
12. Andrew, A.: Another efficient algorithm for convex hulls in two dimensions. Information Processing Letters **9** (1979) 216 – 219
13. Kovács, L., Kovács, A., Utasi, A., Szirányi, T.: Flying target detection and recognition by feature fusion. SPIE Optical Engineering **51** (2012)
14. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: Conference on Computer Vision and Pattern Recognition (CVPR). (2012)
15. King, D.E.: Dlib-ml: A machine learning toolkit. Journal of Machine Learning Research **10** (2009) 1755–1758
16. Kuhn, H.: The Hungarian method for the assignment problem. Naval Research Logistic Quarterly **2** (1955) 83–97