Dynamic 3D environment perception and reconstruction using a mobile rotating multi-beam Lidar scanner

Attila Börcs, Balázs Nagy and Csaba Benedek

Abstract In this chapter we introduce cooperating techniques for environment perception and reconstruction based on dynamic point cloud sequences of a *single* rotating multi-beam (RMB) Lidar sensor, which monitors the scene either from a moving vehicle top or from a static installed position. The joint aim of the addressed methods is to create 4D spatio-temporal models of large dynamic urban scenes containing various moving and static objects. Standalone RMB Lidar devices have been frequently applied in robot navigation tasks and proved to be efficient in moving object detection and recognition. However, they have not been widely exploited yet in video surveillance or dynamic virtual city modeling. We address here three different application areas of RMB Lidar measurements, starting from people activity analysis, through real time object perception for autonomous driving, until dynamic scene interpretation and visualization. First we introduce a multiple pedestrian tracking system with short term and long term person assignment steps. Second we present a model based real-time vehicle recognition approach. Third we propose techniques for geometric approximation of ground surfaces and building facades using the observed point cloud streams. This approach extracts simultaneously the reconstructed surfaces, motion information and objects from the registered dense point cloud completed with point time stamp information.

1 Introduction

Vision based perception of the surrounding environment has a major impact in robotics research with many prominent application areas such as autonomous driving, visual surveillance and virtual city modeling. Outdoor laser scanners, such as Lidar mapping systems have particularly become important tools for gathering da-

Authors are with the Distributed Events Analysis Research Laboratory, Institute for Computer Science and Control (MTA SZTAKI), Hungarian Academy of Sciences, H-1111 Kende utca 13-17, Budapest, Hungary e-mail: firstname.lastname@sztaki.mta.hu.

ta flow for these tasks 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 contrast to conventional camera systems *e.g.* they are highly robust against illumination changes or weather conditions, and they may provide a larger field of view (FoV).

Rotating multi-beam Lidar systems (RMB Lidar) provide a 360° FoV of the scene, with a vertical resolution equal to the number of the sensors, while the horizontal angle resolution depends on the speed of rotation. Each laser point of the output point cloud is associated with 3D spatial coordinates and an intensity value of the laser reflection which is related to the material and surface properties of the target point. The Velodyne HDL-64E RMB Lidar sensor operates with 64 vertically aligned laser transmitters and receivers, and it is able to capture point cloud sequences with a spatial radius of 150m and temporal frame-rate up to 20Hz. Due to its scanning frequency, this equipment is highly appropriate for analyzing moving objects in large outdoor scenes. However, a single scan is quite sparse, consisting of around 65K points with a radius of 120 meters, moreover we can also observe a significant drop in the sampling density at larger distances from the sensor and we also can see a ring pattern with points in the same ring much closer to each other than points of different rings (see Fig. 1 and 2).

In this chapter, we present various algorithms for automated environment perception from three different application fields:

- 3D people surveillance
- Vision of autonomous vehicles
- Large scale urban scene analysis and reconstruction

As the common point, we rely on measurement sequences of a single RMB Lidar sensor in each case. However, the circumstances and the challenges are significantly different in the three scenarios.

A possible *surveillance configuration* of the RMB Lidar sensor is shown in Fig. 1: the sensor is installed in a *fixed position*, and monitors a scene with several mov-

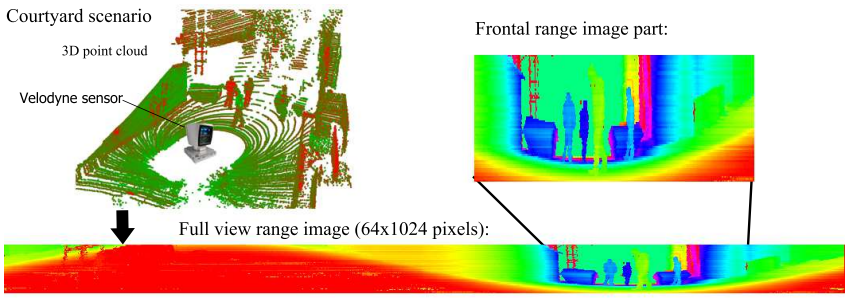


Fig. 1 Surveillance scenario configuration with the Velodyne HDL-64E RMB-Lidar sensor: point cloud and range image representation of a time frame from the measurement sequence

(a) Horizontal Lidar configuration, and a sample frame from a street scan

(b) Tilted measurement configuration and a sample frame from a building facade

Fig. 2 Car mounting configurations of a RMB Lidar sensor and the recorded measurements

ing people in a compact outdoor environment, such as a courtyard or a small square. Since instant system response is needed in cases of unauthorized entries or suspicious person behavior, the algorithms must work *quasi real time*, i.e. at most a few seconds delay is admitted between the occurred event and the warning sent to the operator.

In the vision modules of *autonomous vehicles*, *strict real time* response (within fraction of a second) is necessary based on data streams of a *moving Lidar sensor* which is usually mounted onto the top of the vehicle (see Fig. 2(a)). From the point of view of processing, the main tasks are self localization and obstacle avoidance on the streets (Levinson et al, 2007; McNaughton et al, 2011). While the focus is on perception speed and reliability, the semantic depth of scene exploration and object classification can be rather limited, users do not expect from an autonomous car to gather complex environmental information such as traffic counting or road quality assessment.

The third topic is related to *large scale dynamic urban scene analysis*, where the Lidar is put again to a *moving car's top*, but *offline* data processing and information mining is admitted. Here the main goal is to simultaneously map the observed environment into 3D models, and extract various kinds of information from the dynamic scene, which can be exploited in different application fields, such us 4D virtual city reconstruction, monitoring various public premises, surveys of road marks and traffic signs, urban green area estimation. Depending on our exact aims, we can collect street measurements with two different sensor mounting configurations. By fixing the Lidar horizontally on the top of the vehicle (see Fig. 2(a)), the recorded data

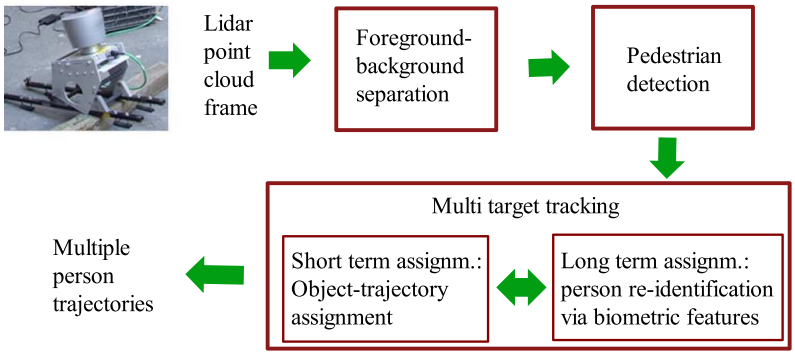


Fig. 3 3D people surveillance: workflow of the multi-pedestrian tracking framework

sequence is appropriate for monitoring the ground and various street objects such us cars, lamp posts traffic signs etc. On the other hand, the tilted configuration enables the scanning and subsequent reconstruction of building facades (Fig. 2(b)).

In this chapter we give an overview on recent results from the above three topics, by using in each scenario the same Velodyne HDL-64E RMB Lidar sensor available at MTA SZTAKI in Budapest, Hungary. The measurement of the RMB Lidar within a given time frame can be represented as a *point cloud* p of $R \cdot c$ points, where R is equal to the number of the vertically aligned laser transmitters providing concurrent distance measurements (here $R = 64$) and c is the number of *point columns*, i.e. the number of time samples collected during a whole 360° turn of the rotating sensor ($c \approx 1024$ was observed by 20 Hz rotation). Apart from the geometric parameters, each point p receives an intensity value, which is related to the echo strength from a given direction. In Fig. 1 and 2 sample raw Lidar frames are displayed, where the green-to-red point coloring is based on the intensity. In Sections 2-4, we introduce the three applications one after another. The measurement of the RMB Lidar within a given time frame can be represented as a *point cloud* p of $R \cdot c$ points, where R is equal to the number of the vertically aligned laser transmitters providing concurrent distance measurements (here $R = 64$) and c is the number of *point columns*, i.e. the number of time samples collected during a whole 360° turn of the rotating sensor ($c \approx 1024$ was observed by 20 Hz rotation). Apart from the geometric parameters, each point p receives an intensity value, which is related to the echo strength from a given direction. In Fig. 1 and 2 sample raw Lidar frames are displayed, where the green-to-red point coloring is based on the intensity. In Sections 2-4, we introduce the three applications one after another.

2 3D people surveillance

Moving people detection, localization and tracking are important issues in intelligent surveillance systems, as parts of person counting, activity recognition or abnormal event detection modules. However, these tasks are still challenging in crowded outdoor scenes due to uncontrolled illumination conditions, irrelevant background motion, and occlusions caused by various moving and static scene objects. Person re-identification is a fundamental task both for connecting the erroneously broken trajectories of the short term tracker module, and for identifying people who temporarily leave the Field of View (FoV) and re-appear later. Range sensors have recently started to be used in various surveillance applications. Time-of-Light (ToF) cameras (Schiller and Koch, 2011) record depth image sequences over a regular 2D pixel lattice, where established image processing approaches, such as Markov Random Fields (MRFs) can be adopted for smooth and observation consistent segmentation and recognition. However, such cameras have a limited FoV, which can be a drawback for surveillance and monitoring applications. In this section, we focus on the sensor configuration of Fig. 1, where the RMB Lidar device records 360°-view-angle range data sequences of irregular point clouds.

2.1 Foreground-background separation

To separate dynamic foreground from static background in a range data sequence, a probabilistic approach of Benedek (2014) is applied. We project first the irregular point cloud to a cylinder surface yielding a depth image on a regular lattice, and perform the segmentation in the 2D range image domain (see Fig. 1). We model the statistics of the range values observed at each pixel position as a Mixture of Gaussians and update the parameters similarly to the standard approach of Stauffer and Grimson (2000). The background is modeled by the Gaussian components with the highest weight values in the mixture, and outlier detection enables the extraction of the possible motion regions. However, by adapting the above scheme, we must expect several spurious effects, caused by the quantization error of the discretized view angle and background flickering, e.g., due to vegetation motion. These effects are significantly decreased by a dynamic MRF model introduced in Benedek (2014), which describes the background and foreground classes by both spatial and temporal features. Since the MRF model is defined in the range image space, the 2D image segmentation must be followed by a 3D point classification step by resolving the ambiguities of the 3D-2D mapping with local spatial filtering. Using a spatial foreground model, we remove a large part of the irrelevant background motion which is mainly caused by moving tree crowns.

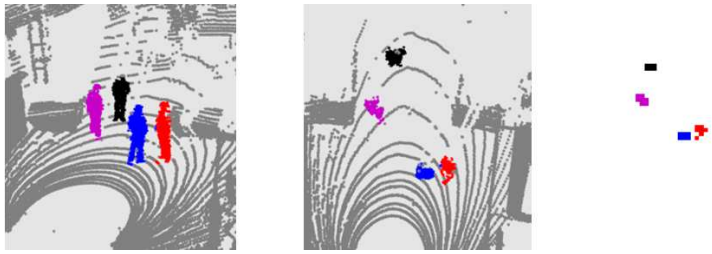


Fig. 4 Pedestrian separation. Left: side view of the segmented scene, centered: top view, right: projected blobs in the image plane

2.2 Pedestrian detection and multi-target tracking

In this section, we introduce the pedestrian detection and tracking module of the system. The input of this component is the RMB Lidar point cloud sequence, where each point is marked with a segmentation label of foreground or background, while the output consists of clusters of foreground regions so that the points corresponding to the same person receive the same label over the sequence. We also generate a 2D trajectory of each pedestrian.

First, the point cloud regions classified as foreground are clustered to obtain separate blobs for each moving person. A regular lattice is fit to the ground plane and the foreground regions are projected onto this lattice. Morphological filters are applied in the image plane to obtain spatially connected blobs for different persons. Then the system extracts appropriately sized connected components that satisfy area constraints determined by lower and higher thresholds. The center of each extracted blob is considered as a candidate for foot position in the ground plane. Note that connected pedestrian shapes may be merged into one blob, while blobs of partially occluded persons may be missed or broken into several parts. Instead of proposing various heuristic rules to eliminate these artifacts at the level of the individual time frames, a robust multi-tracking module has been developed, which efficiently handles the problems at the sequence level.

The pedestrian tracking module combines Short-Term Assignment (STA) and Long-Term Assignment (LTA) steps. The STA part attempts to match each actually detected object candidate with the current object trajectories maintained by the tracker, by purely considering the projected 2D centroid positions of the target. The STA process should also be able to continue a given trajectory if the detector misses the concerning object in a few frames due to occlusion. In these cases the temporal discontinuities of the tracks must be filled with estimated position values. On the other hand, the LTA module is responsible for extracting discriminative features for re-identification of objects lost by STA due to occlusion in many consecutive frames or leaving the FoV. For this reason, lost objects are registered to an archived object list, which is periodically checked by the LTA process. LTA should also recognize if a new person appears in the scene, who was not registered by the tracker beforehand.

2.2.1 Short Term Assignment (STA) module:

The STA module receives the measured ground plane positions and for each frame it iterates three basic operations, namely, data assignment, Kalman filter correction and Kalman filter prediction. The assignment operation matches the currently measured candidate positions to the registered object trajectories with using the Hungarian algorithm (Kuhn, 1955). Then the estimated object positions are corrected and, finally, predictions for the subsequent positions are made and fed back to the assignment procedure. The algorithm can handle false positives, broken trajectories as well as tracks starting and terminating within a sequence.

By a given time frame, an object is marked as *Active* target, if its trajectory can be continued with one of the actually measured positions. Due to expected occlusion and noise effects, trajectories which cannot be matched with a new observation are not terminated immediately. They are marked first as *Inactive* tracks, which also participate in the STA process in the upcoming measurement iterations for at most a few seconds time window. The final step of the trajectory update is to make the Kalman prediction for the next point of each track, which can be used for measurement assignment in the next time frame (details are presented in (Benedek, 2014)).

2.2.2 Long Term Assignment (LTA) features

In (Benedek, 2014) two static point cloud descriptors were adopted for person re-identification. *First*, the authors have observed that since clothes of people consist of various materials, the intensities obtained by the RMB Lidar sensor exhibit different statistical characteristic for different people. For each tracked target, an intensity histograms has been collected over at least 100 frames, and used as feature for comparison. Experimental evidences have shown that the Bhattacharyya distance of the *normalized* intensity histograms for two object samples efficiently indicated whether the candidates correspond to the same person or not.

As a *second* feature, the height of the person were estimated, by taking the elevation difference of the highest and lowest object points at each time frame, and creating a histogram from the measured height instances. The height estimate of the person has been taken as the peak value of the *actual height histogram* over several frames.

Since both features have been derived by temporal feature statistics, a newly appearing object must enter first an *Initial* phase, where the long-term histograms are accumulated. After a given number of frames, one can execute the LTA process which marks the object as *Identified*. We accept a long term target match only if both the intensity and the height difference features show relevant similarity. Pedestrians unsuccessfully matched to any archived objects by LTA receive a new unique identifier.

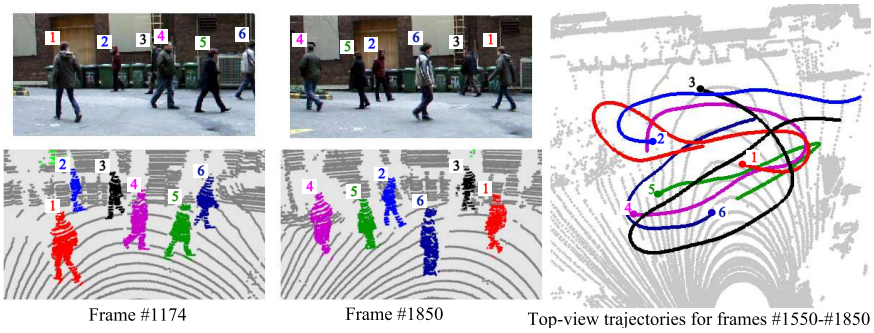


Fig. 5 Results of pedestrian separation and tracking in the `Winter2` Lidar sequence. Video images (in the top) were only used for validation of tracking and re-identification.

2.2.3 Tracking process

Based on the previously introduced STA and LTA modules, the tracking process is realized by a finite-state machine with the following states: *Init-Active*, *Init-Inactive*, *Identified-Active*, *Identified-Inactive*, *Deleted* and *Archieved*. The names of the first four states encode, if a given actually tracked object is currently *Active* or *Inactive* according to the STA module, and if it is already *Identified* or is yet in the *Init*ialization phase of LTA. Transitions between the corresponding *Active* and *Inactive* states are controlled by the STA module, depending on the success of matching the existing trajectories with actual measurements. *Identified* objects which are *Inactive* for more than T_{SIL} seconds (used $T_{\mathrm{SIL}} = 3$ sec.) are moved to the archive list: *Archived* objects do not participate in the STA process, but they can be re-activated later by LTA. Objects spending T_{SIL} seconds in the *Init-Inactive* state are marked as *Deleted*, and excluded from the further investigations during the tracking process. These deleted trajectories usually correspond either to measurement noise, or they are too short to provide us reliable descriptors for later LTA matching.

The LTA identification process can be applied for objects which have spent in the *Init-Active* state at least 8 seconds, which proved to be an appropriate time interval frame for the consolidation of the LTA features. If a match is successful with an archived object, the trajectories of the new and matched objects are merged with interpolating the missing trajectory points. Then the LTA-matched *Archived* object is moved to the *Identified-Active* state, and the new object is *Deleted* to prevent us from duplicates. On the other hand if the LTA match fails, the new object steps to the *Identified-Active* state with keeping its identifier.

2.3 Evaluation

We have evaluated our method in various outdoor Lidar sequences containing multi-target scenarios recorded in the courtyard of our institute. Since the sequences were

recorded in different seasons (`Summer`, `Winter` and `Spring`), we could also investigate the robustness of the used point cloud features against the effects on different clothing styles (such as winter coats or t-shirts).

The sequences contain 4-8 people walking in a $220m^2$ area FoV in 1-15m distances from the Lidar. The rotation speed was set from 15Hz to 20Hz. In the background, heavy motion of the vegetation make the accurate classification challenging. We have also recorded the test scenarios with a standard video camera *only* for verification of the tracking and re-identification process. The advantage of using sequences from different seasons was that we could test the robustness of the approach versus seasonal clothing habits (winter coats or T-Shirts) and illumination changes.

We have verified the multiple people tracking and re-identification modules by counting the correct and incorrect trajectory matches during the whole observation periods. For quantitative evaluation of the tracking process the output trajectories of the system were verified by manual observes watching the point cloud sequences and the recorded videos in parallel.

As evaluation metrics, we counted the following events (see results in Table 1):

- *STA trans. num*: number of all *Inactive*→*Active* state transitions during the tracking process, i.e. the number of events, when the Short-Term Assignment (STA) module can continue a track after the object had been occluded for a couple of frames (counted automatically).
- *STA trans. error*: number of erroneous track assignments by the STA module (counted manually).
- *LTA trans. num*: number of *Archived*→*Identified* state transitions during the tracking process, i.e. the number of events, when the Long-Term Assignment (LTA) module can recognize a previously archived and re-appearing person (counted automatically).
- *LTA trans. error*: number of erroneous person assignments by the LTA module (counted manually).

The – altogether seven – surveillance sequences listed in Table 1(b) imply varying difficulty factors for the multi-target tracking process. First, we calculated the *Av*erage *people* number *per frame* (4th column) among the frames of the Lidar sequence, which contain at least two pedestrians. Higher people density results in more occlusions, thus usually in increasing *STA trans. num*, which means chal-

Table 1 Person tracking evaluation on seven surveillance test sequences. STA: Short-Term Assignment, LTA: Long-Term Assignment

Set name	**Seq. num.**	**Frame num.**	**People num.**	**Av peopl. per frame**	**STA trans. num (error)**	**LTA trans. num (error)**	**Processing speed (fps)**
`Summer`	3	4922	4	3.61	131 (0)	1 (0)	13.03
`Winter`	2	6074	4-6	3.49	200 (0)	19 (0)	12.81
`Spring`	2	4999	6-8	6.95	343 (1)	33 (5)	12.62

lenges for the STA module. On the other hand, the total number of people (4-8) and the *LTA trans. num* affect the LTA re-identification process. Table 1 confirms that tracking errors only occur by the `Spring` sequence, where the number (and density) of people is the highest.

Fig. 5 displays two sample frames from the `Winter/2` sequence. Between the two selected frames, all pedestrians left the FoV, therefore a complete re-assignment should have been performed by the LTA module. Note that even with applying Kalman filtering, the resulted raw object tracks are quite noisy, therefore, we applied a 80% compression of the curves in the Fourier descriptor space (Zhang and Lu, 2002), which yields the smoothed tracks displayed in Fig 5, right.

An important feature of the proposed system is the nearly real time performance with processing 15 Hz Lidar sequences. The last column of Table 1 lists the measured processing speed on the different test sets. Compared with fps values of Table 1(a), we can conclude that the most expensive part of the process is foreground-background segmentation (in itself 15-16 fps), since the complete workflow including foreground detection, pedestrian separation and tracking operates with 12-13 fps.

3 Real time vehicle detection for autonomous cars

In the vision modules of self-driving cars or driving assistance systems, *real time* response is necessary based on data streams of a *moving Lidar sensor* which is usually mounted onto the top of the vehicle. These 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.

A number of automatic point cloud analysis methods have been proposed in the literature for RMB Lidar streams. These approaches mainly focus on research towards real time point cloud classification for robot navigation and quick intervention. Douillard et al (2011) presents a set of clustering methods for various types of 3D point clouds, including dense 3D data (e.g. Riegl scans) and sparse point sets (e.g. Velodyne scans), where the main goal is to approach close to real-time performance. The object recognition problem from a segmented point cloud sequence is often addressed with machine learning techniques relying on training samples. A boosting framework has been introduced in (Teichman et al, 2011) for the classification of arbitrary object tracks obtained from the Lidar streams. This step needs accurately separated obstacles or obstacle groups as input, but it deals neither with the context of the objects nor with large surface elements such as wall segments. In (Xiong et al, 2011) the authors model the contextual relationships among the 3D points, and train this procedure to use point cloud statistics and learn relational information, e.g. tree-trunks are below vegetation, over fine and coarse scales. This point cloud segmentation method shows its advantage on the classes that can provide enough training samples, however domain adaption remains a difficult chal-

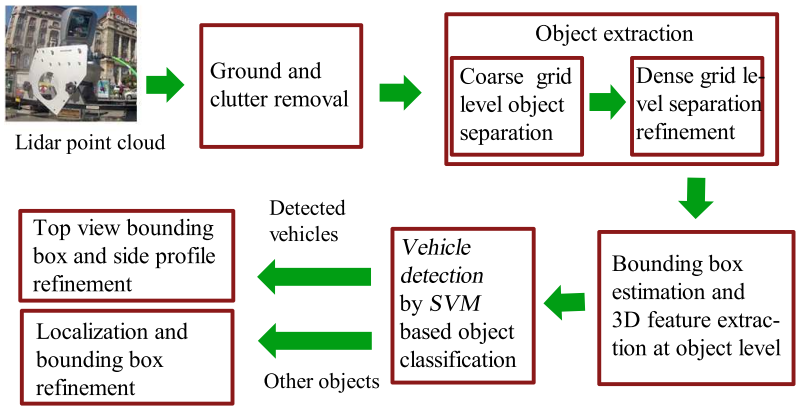


Fig. 6 Workflow of the proposed real time vehicle detection method

lenge. (Quadros et al, 2012) presented a feature called *the line image* to support object classification that outperforms the widely used NARF descriptor but requires a computationally expensive principal component analysis (PCA) calculation.

In this section we present a model-based real-time approach for vehicle detection and extraction from continuously streamed Lidar point clouds, which are captured in challenging urban scenarios (Börcs et al, 2014a,b). The workflow of the proposed method is demonstrated in Fig. 6. The first step is the quick removal of the ground and clutter regions. Thereafter the field objects are extracted by a novel two-level gird based connected component analysis method. This algorithm is able to efficiently separate the objects even if they are close to each other, while it can maintain the real-time performance of the system by processing the unorganized point cloud streams. Our next objective is to recognize the vehicles from the set of object candidates. For this reason, we estimate the top-view 2D bounding boxes of the objects, and obtain different point cloud based 3D features, which offer useful information for vehicle identification. The classification is performed then by a *Support Vector Machine* (SVM). In post-processing the side profile contours of the vehicles are also estimated, while for other objects we provide top view bounding boxes as the outputs.

3.1 Object extraction by point cloud segmentation

Two mainstreams for point cloud segmentation in the literature are 3D point clustering, and 2D lattice based point classification approaches. Point clustering methods work directly in the Euclidean space of measurements, thus they can directly exploit the 3D structure information stored in the Lidar data. These approaches use usually a 3D space partitioning structure for quick data access. In case of dealing

with RMB Lidar scans, adaptive partitioning techniques should be adopted, such us octrees or kd-trees, due to the highly inhomogeneous density characteristics of the point clouds. Moreover, since the sensor is moving, these tree structures should be independently built for each time frame, causing a notable computational overload.

For the above reason, we decided to follow the 2D lattice based approach. Here the points are projected to a horizontal plane where a regular 2D grid is defined, so we simplify the 3D point neighborhood search problem to investigating regular pixel neighborhoods. However, in this case we may loose a significant amount of information by point projection and the grid discretization. To overcome these limitations, we keep the height information of each projected Lidar point assigned to the lattice and develop a two-level (coarse and fine) 2D grid structure to achieve very quick but accurate object separation.

3.1.1 Ground and clutter removal

Ground detection and initial noise filtering is achieved by a standard grid based approach, using a regular 2D grid fitted to the ground, with rectangle width between 50cm and 80cm. First, local point cloud density is calculated for each cell to extract points of the *clutter* class, which marks the sparse cells. The exact density threshold τ_d depends on the sensor's revolving speed, we used 4-8 points for a given cell.

The next step is terrain modeling. Planar *ground* models are frequently adopted in the literature relying on robust plane estimation methods such as RANSAC. However, in the considered urban scenes we experienced significant elevation differences (often up to a few meters) between the opposite sides and central parts of the observed roads and squares. In these cases, planar ground estimation yields significant errors in the extracted object shapes, e.g. bottom parts can be cut off, or the objects may drift over the ground. *On the contrary*, we apply a locally adaptive terrain modeling approach. As a first evidence, we can notice that in the ground cells the differences of the observed elevation values are small.

Therefore we can perform an initial classification, where each cell s is classified either as ground candidate ($\mathbf{1}_G(s) = 1$) or as undefined region ($\mathbf{1}_G(s) = 0$) by a straightforward thresholding:

$$\mathbf{1}_G(s) = 1 \text{ iff } \big(y_{\max}(s) - y_{\min}(s) < \tau_{\text{gr}}\big),$$

where we used τ_{gr} =25cm. 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. This preliminary map can only be considered as a coarse estimation of the ground, since cells of flat car roof or engine hood regions may be erroneously classified as ground, for example. However, these outlier cells can be efficiently eliminated by spatial filtering. With denoting by N_s^v the $v \times v$ neighborhood of s, and $\gamma_G(s) = \sum_{r \in N_s^v} \mathbf{1}_G(s)$, we can obtain a terrain model of scene:

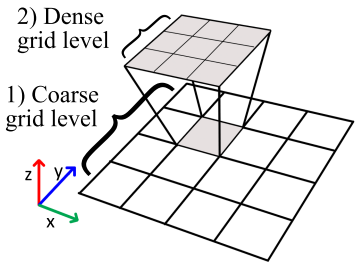


Fig. 7 Visualization of our *hierarchical grid model* data structure - *(bottom)* the coarse grid level: the 3-D space coarsely quantized into 2-D grid cells, *(top)* the dense grid level: each grid cell on the coarse level subdivided into smaller cells.

$$y_{\mathrm{gr}}(s) = \begin{cases} \frac{1}{\gamma_G(s)} \cdot \sum_{r \in N_s^v} \hat{y}(s) \cdot \mathbf{1}_G(s) & \text{if } \gamma_G(s) > 0 \\ \text{undefined} & \text{otherwise.} \end{cases}$$

where $y_{\mathrm{gr}}(s)$ is the estimated ground-elevation value at cell s. The $y_{\mathrm{gr}}(s)$ feature can be efficiently calculated by deriving the integral images of the $\hat{y}(.)$ and $\mathbf{1}_G(.)$ maps. We used here a large neighborhood ($v = 17$ for cell maps with a size around 400×300). Finally a cell is classified as ground cell iff $\mathbf{1}_G(s) = 1$ and $\hat{y}(s) - y_{\mathrm{gr}}(s) <$ 20 cm.

3.1.2 Object separation by a two-level grid based model

After removing the ground and clutter regions, an estimation for the scene objects can be obtained by connected component extraction from the remaining point cloud parts. For maintaining the quick computational speed of the process, similarly to Sect. 3.1.1 we use again a 2D grid based approach. The basic idea is that we segment the 2D cell map into connected regions, thereafter we backproject the obtained cell-labels to the original point cloud, and assume that the points with same labels correspond to the same object. In the grid we consider a cell as *foreground* (i.e. object) cell, if it contains the projection of least τ_d object points. We also create an elevation map by assigning to each *foreground* cell the maximum of the corresponding point height. The cell map segmentation is implemented by a constrained connected component labeling process, where two neighboring *foreground* cells can only be merged, if the difference between their elevation map values are also lower than a threshold (used 40cm). In addition, by the scanning of the grid we ensure that only convex object regions can be extracted, i.e. a negative elevation slope cannot be followed by a positive one within an object neither in the *x* nor the *y* direction.

As the main limitation of the standard grid based segmentation, the optimal cell size is always obtained by a trade-of. Using larger cells, close objects may be merged into the same extracted blob, while smaller cells may yield that many objects break into several pieces. As a solution, a hierarchical two-level grid structure has been proposed in (Börcs et al, 2014b), where each cell of the *coarse* (upper) level is divided into *smaller* sub-cells at the fine level, as shown in Fig. 7. The rough object estimation is performed at the coarse grid level, but the extracted super object blobs may be cut into many smaller objects, considering the fine level cells. The technical

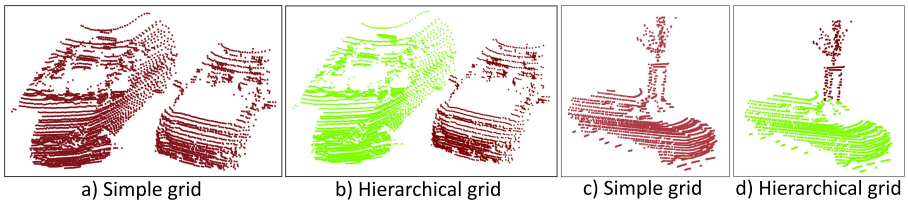


Fig. 8 Object separation for a case of nearby objects. Comparison of the *Simple Grid Model* Fig. a), c) and the *Hierarchical Grid Model* Fig. b), d).

details of the approach and parametrization can be found in (Börcs et al, 2014b). Sample results for objects merged at coarse level, but correctly separated by the hierarchical model are demonstrated in Fig. 8.

3.2 Object level feature extraction and vehicle recognition

The input of the vehicle detection step are the object point clouds extracted in Sec. 3.1, and the task can be formulated as binary blob classification with *vehicle* and *non-vehicle* classes. he proposed vehicle model is based on three different features (Börcs et al, 2014a).

First a new 2D bounding box fitting method has been proposed for the top-view projection of the objects. From an accurate bounding box, relevant volume and side ratio parameters can be derived, which are used by the proposed object classifier. However, the bounding box extraction task proved to be highly challenging using the RMB Lidar point clouds. As shown in Fig. 9(a), conventional Principal Component Analysis (PCA) based solutions usually fail here, since only the object side facing the sensor is clearly visible. 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. For this reason, we developed a new box estimation method, which is based on the convex hull of the projected object shape, and it attempts to detect a corner with connecting orthogonal lines within the

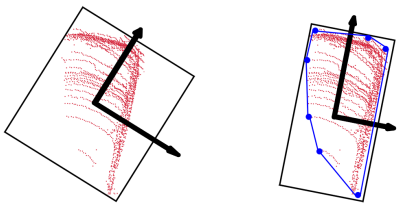


Fig. 9 Demonstrating the limitations of PCA based bounding box approximation, and the advantages of the proposed convex hull based bounding box fitting technique on the top-view projection of a selected vehicle in the point cloud

contour, which is used to fit a highly accurate box to the point set (Fig. 9(b)). The steps of the algorithm are as follows (see also Fig. 10):

- 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.

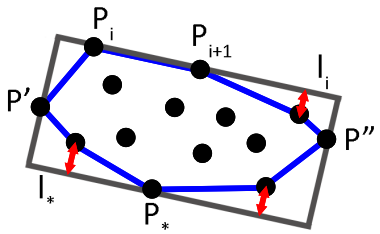


Fig. 10 Demonstration of the fast 2-D bounding box fitting algorithm for the convex hull of the top-view object projection (the bounding box is shown marked by gray color

For defining the *second* feature, we 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. 11 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. 11 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.

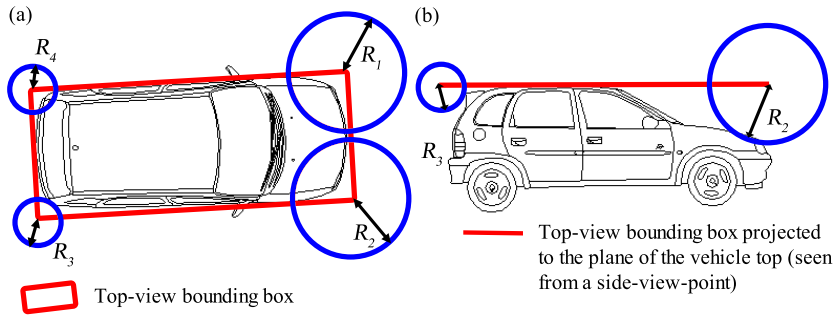


Fig. 11 Demonstration of the principal curvature feature with 3-D spheres

The *third* feature is based on 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 vertical 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 (Kovács et al, 2012).

Finally our joint feature vector for vehicle detection is composed by fusing the following 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.

In the next step, we considered the above defined eight dimensional feature vector, and a SVM classifier has been trained for a 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, several positive vehicle samples, and negative samples (different street furniture and other urban objects) have been selected from the KITTI Vision Benchmark Suite (Geiger et al, 2012) and used for the training of the classifier.

3.3 Evaluation of real-time vehicle detection

We evaluated our method various 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 (Geiger et al, 2012). We have compared our *Model-based approach* to a reference solution, which uses a simple occupancy grid representation for foreground separation, and applies Principal Component Analysis (PCA) based features for object classification (Himmelsbach et al, 2008).

Qualitative results of our proposed model on four sample frames are shown in Fig. 12. During the quantitative evaluation, we verified the proposed method and the reference PCA based technique on 2076 (Budapest) + 614 (KITTI) vehicles, by measuring the object level F-rate of the detection (harmonic mean of precision and recall). We have also compared the processing speed of the two methods in frames per seconds (fps). The numerical performance analysis is given in Table 2. 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 (Budapest data), 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. .

Table 2 Numerical comparison of the detection results obtained by the Principal Component Analysis based technique (Himmelsbach et al, 2008) and the proposed *Model-based framework*. The number of objects (NO) are listed for each data set, and also in aggregate.

Point Cloud Dataset	NO	PCA based ap.		Prop. Model-based ap.	
		F-rate(%)	Avg. processing speed (fps)	F-rate(%)	Avg. processing speed (fps)
Budapest Datasets	2076	69	13	88	22
KITTI Dataset	614	62	14	78	25

4 Large scale urban scene analysis and reconstruction

Understanding of large dynamic scenes and 3D virtual city reconstruction have been two research fields obtaining great interest in the recent years. Although these tasks have usually been separately handled, connecting the two modalities may lead us to realistic 4D video flows about large-scale real world scenarios, which can be viewed and analyzed from an arbitrary viewpoint, can be virtually modified by user interaction, resulting in a significantly improved visual experience for the observer. However, the proposed integration process faces several technical and algorithmic

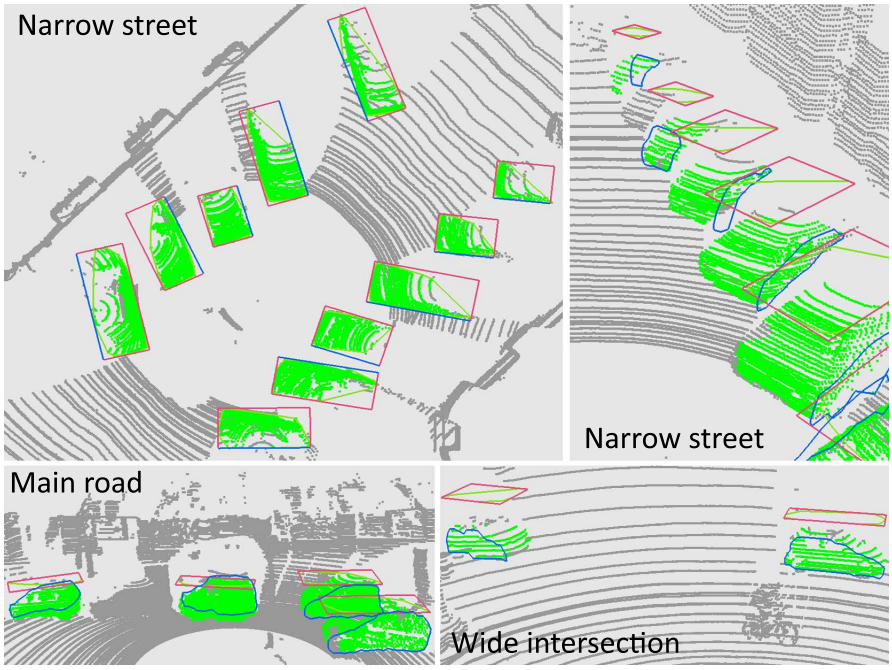


Fig. 12 Qualitative results of vehicle detection with displaying the top-view bounding boxes (by red) and the side view concave hulls (blue) extracted by the algorithm

challenges. On one hand, comprehensive scene interpretation, object clustering & long-term tracking and event recognition from optical videos or 2.5D range image sequences are still challenging problems, in particular if the measurements are provided by moving sensors. On the other hand, virtual 3D city visualization needs dense registered information extracted from the scene, enabling the realistic reconstruction of fine details of building facades, street objects etc.

RMB Lidar systems may offer efficient solutions for the task due to their high temporal refreshing rate, but only a limited information can be extracted from the individual time frames which are sparse and their density is notably inhomogeneous. For the above reason, by using RMB sensors in complex situation interpretation and scene visualization tasks, merging multiple measurement frames into a joint high resolution point cloud is an inevitable step.

4.1 Multiframe point cloud processing framework

In this section we simultaneously deal with the recognition and mapping issues without relying on any additional sensor information apart from the RMB Lidar stream (Józsa et al, 2013). The proposed method consists of six main steps, as shown in

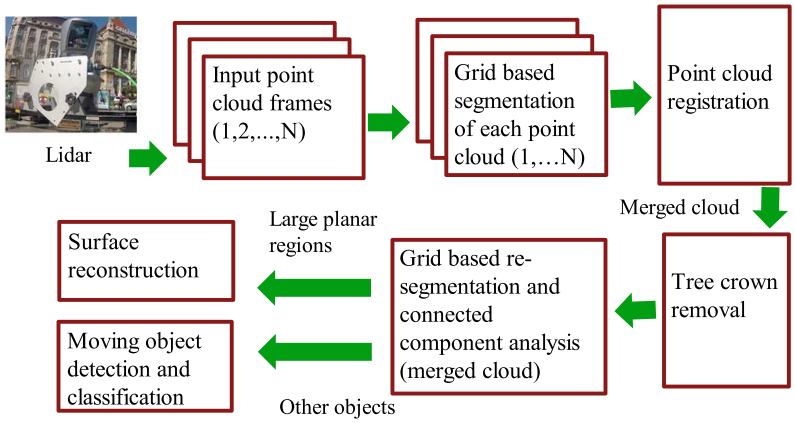


Fig. 13 The workflow of the proposed algorithm

Fig. 13. *First*, the individual Lidar point cloud scans are segmented into different semantic regions. *Second*, the Lidar frames are automatically registered, i.e. transformed to a common coordinate system, with preserving the original time stamp for each point. *Third*, vegetation is detected and the initial segmentation is refined by exploiting features from the merged point cloud. *Fourth*, large planar regions (e.g. facades) and other street objects are separated with a flood fill based step. *Fifth*, large planar regions are triangulated, while *sixth*, street objects are classified either as static or moving entities, and trajectories of moving objects are extracted.

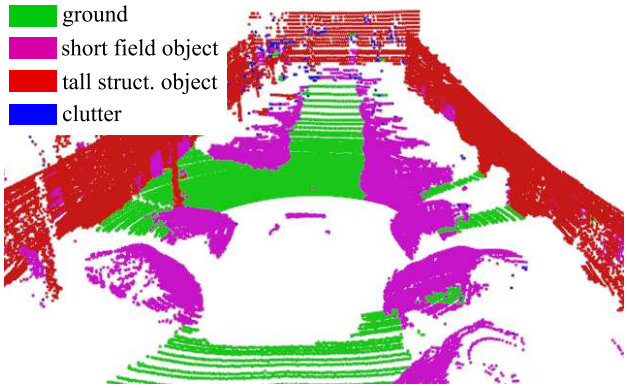


Fig. 14 Segmented frame of the Velodyne point cloud stream. Note: figures of this paper are best viewed in color print.

4.1.1 Point cloud segmentation

The segmentation process assigns to each measured point a class label from the following set: (i) *clutter* (ii) *ground*, (iii) *tall structure objects* (walls, roofs, lamps posts, traffic lights etc.), (iv) *short street objects* (vehicles, pedestrians etc.) and (v) *vegetation*. In this section we address the discrimination of the first four classes, while vegetation will be only removed after the point cloud registration step.

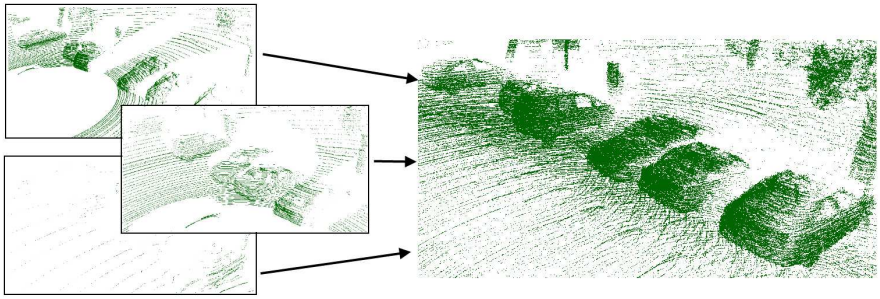


Fig. 15 Image sequence showing the registration process. Right side image contains 20 registered scans

The segmentation process follows a single grid based approach, where the ground and clutter detection step is achieved in the same way, as presented in Sec. 3.1.

A cell corresponds to *tall structure objects*, if either the difference of the maximal and minimal elevations of the included points is larger than a threshold (used 310 centimeters), or the maximal observed elevation is larger than a predefined value from the sensor (used 140 centimeters). The second criterion is needed for dealing with objects standing on a lower point of the ground.

The rest of the cells are assigned to class *short street objects* like vehicles, pedestrians, short road signs, line posts etc. These entities can be either dynamic or static, which attribute can only be determined later after further, more complex investigation of the point cloud sequence.

After classifying the cells of the 2D cell map, we have to assign a class to each point of the 3D point cloud as well. Usually, each point p obtains the label of its parent cell s. However, for cells contain both ground and tall (or short) object regions, the classification yields that ground segments are attached to the object blobs, showing a typical 'carpet' pattern. Therefore, we also cluster p as ground, if although its cell s has any kind of object label, s is neighbored with a ground cell r and $|y(p) - \hat{y}(r)| < 15$ cm.

4.1.2 Point cloud registration

Although a single RMB Lidar scan has a large amount of points, it covers a large area, and the resolution is sufficiently good only within few meters of distance. Though the device has a sensing distance of more than 100 meters, the measurements at more than 15 meters of distance are too sparse for many detection or surface reconstruction algorithms.

In this section, we propose a method for automatic registration of the consecutive Lidar scans, yielding dense and detailed point clouds of large street scenes. Although various established techniques do exist for point cloud registration, such as Iterative Closest Point (ICP) (Zhang, 1994) and Normal Distribution Transform (NDT) (Magnusson, 2009), these methods fail, if we try to apply them for the raw Velodyne Lidar point clouds for two reasons:

- All points reflected from moving objects appear as outliers for the matching process, and since in a crowded street scene we expect a large number of moving objects, many frames are erroneously aligned.
- Due to the usual concentric circle patterns on the ground (see Fig. 14), even the static points may mislead the registration process. However, we have also observed that the point density is quite uniform in local wall regions which are perpendicular to the ground.

Our key idea is to utilize the point classification result from the previous section to support the registration process. As input of the registration algorithm, we only use the points segmented as *tall structure objects*. We expect that in majority, these points correspond to stationary objects (such as buildings), thus they provide stable features for registration. The NDT algorithm was applied to match the selected regions of the consecutive frames of the point cloud, since it proved to be efficient with the considered data and it is significantly quicker than the ICP.

The NDT approach divides the space into cubes and for each cube, it calculates a local probability density function (*pdf*) to describe that cube, so that each *pdf* can be seen as an approximation of the local surface, describing the position of the surface as well as its orientation and smoothness (Magnusson, 2009). For the registration step, it uses Newtons optimization method to find the rotation and translation between the two point clouds, searching for the best match between the *pdf*s of the two scans. This method is robust to outliers.

After calculating the optimal transformation, the whole point cloud of each frame is registered to a joint world coordinate system. This step yields a large and dense point cloud about the scene. However, to enable us exploiting the temporal information stored in the Lidar sequence in the further processing steps, we also maintain for each point its original time stamp in the merged cloud. We note that the proposed registration method is able to deal both with the standard forward facing and with tilted configurations of the Lidar sensor when mounted atop of a vehicle. The forward facing configuration is more suitable for road mapping, traffic monitoring, object detection and tracking, while tilted mounting may result in complete models of tall building facades based on the RMB Lidar data, as shown in Fig. 16.

Fig. 16 Sample results on facade approximation based on RMB Lidar data with the proposed approach

4.1.3 Tree crown detection and segmentation refinement

Tree crown detection is a significant step for two reasons. On one hand, vegetation mapping is important for calculating the green area in a city and marking the trees in the reconstructed city models. On the other hand, the removal of the detected vegetation data from the point cloud can help detection algorithms, for example in the case of trees hanging over parking cars. We have developed a tree crown removal algorithm for the merged point cloud, which calculates a statistical outlier detector feature (Rusu and Cousins, 2011) for each point based on the mean distance to the 25 nearest neighbors, and also exploits the intensity channel which is an additional indicator of vegetation, which reflects the laser beam with a lower intensity (see Fig. 17). Thereafter, we also refine the separation of *ground*, *tall* and *short street objects* in the registered cloud, using the classification steps introduced in Sec. 4.1.1.

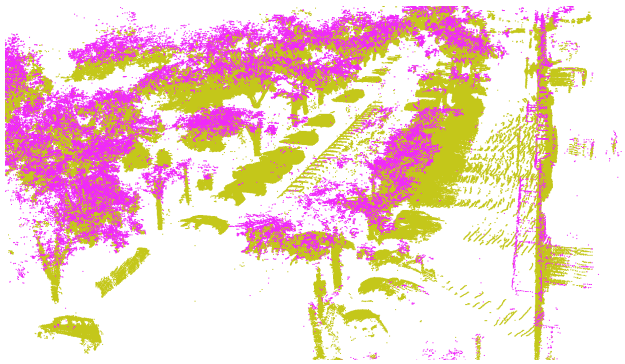


Fig. 17 Tree crown detection (marked with purple).

4.1.4 Object separation with spatio-temporal flood fill

After removing ground and clutter points from the merged and segmented point cloud, the different objects and surface components are separated with flood fill propagation starting from random seed points, which step is repeated until every point receives a unique object label. The proposed algorithm has two key properties. *First*, we separately apply the recursive flood fill steps for point cloud regions of *tall structure objects* and *short street objects*. In this way, pedestrians walking close to walls or lamp posts are efficiently separated from the structure elements. *Second*, since moving objects yield large connected regions in the merged point cloud (Fig. 18), different object blobs may erroneously be connected due to motion. For this reason, when we create the connected components with the flood fill algorithm, we also consider the time stamps of the points: for a given seed point we only assign a neighbor to the same segment, if the distances of both the locations and time stamps are below given thresholds. Point cloud segments with large extent are considered as facade segments and - together with terrain regions - they are transferred to the upcoming surface approximation step. Small connected components of the short object class are excluded from the further investigations.

4.1.5 Surface approximation with triangle meshes

Raw RMB Lidar point cloud frames are not suitable for geometric surface reconstruction due to the low and strongly inhomogeneous density of the individual scans. However, after registering several point clouds against each other with our technique proposed in Sec. 4.1.2, the resolution can be sufficiently high and uniform to create realistic building facade reconstruction. As the car passes by a building, it collects data from several point of view so most of the holes on the walls due to occlusion can be filled in. Also, after concatenating a few dozen scans, the resolution of the data will be significantly higher which results in a precise 3D reconstruction of wall surfaces and more efficient noise reduction also. Fig. 16 displays a triangulated mesh obtained with the Poisson surface reconstruction algorithm (Kazhdan et al, 2006).

4.1.6 Object level analysis

As mentioned in Sec. 4.1.4, the regions of moving objects in the merged point cloud cause blurred object blobs, which should be indicated. Although dynamic regions have generally a lower point density, in our experiments the simple local density-feature proved to be inefficient for motion separation. Instead, we utilized the following blob-based feature: after we extracted the connected blobs of the *short street objects* regions in the merged cloud with flood fill propagation (Sec. 4.1.4), within each blob we separate the points corresponding to the different time stamps and determine their centroids. Assuming that the centroids of the same object follow lines

or curves if the object is moving and stay within a certain region if the object is static (Fig. 18(a)), we can cluster the moving and static object regions as shown in Fig. 18(b).

An important impact of this clustering step is that the static objects can be analyzed henceforward in the merged point cloud, which may provide significant higher level information about the entities, e.g. for recognizing various car types (Fig. 15).

4.2 Experiments

In this section, we present quantitative evaluation of the proposed methods on real urban point cloud streams. First, we show the effectiveness of using our point cloud segmentation algorithm (i.e. *presegmentation*) to support the automatic registration process of the consecutive Lidar frames. Second, we present an object level analysis of the proposed detector using the registered 'spatio-temporal' point clouds.

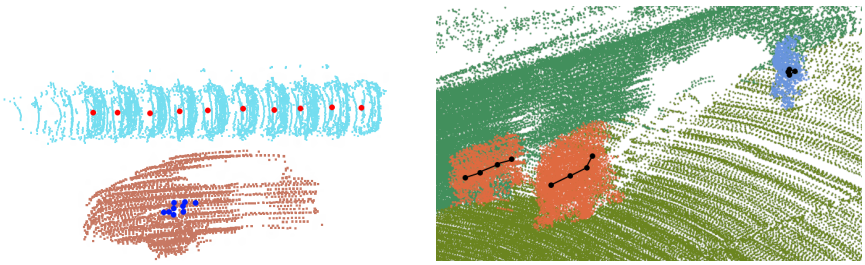
As a quantitative evaluation metrics for the proposed registration algorithm (Sec. 4.1.2), we used the crispness feature of (Douillard et al, 2012). The crispness is calculated on a regularly voxelised point cloud by counting the number of occupied voxels. As the authors assume there, the lower this number, the more crisp the point cloud and in turn the more accurate the alignment.

We compared the results obtained by the proposed method with the presegmentation step (Pre+NDT) to the output of the NDT algorithm applied on the raw Velodyne frames (Raw NDT). Table 3. shows the evaluation results of two scene sets comparing the two methods using a 10cm voxel grid. The scenes were selected in a way that they represent different city-scenarios, including both slow and high speed sensor movement recorded in streets and squares. Speed columns show the overall registration time that was needed to register 10 point clouds. The crispness feature was calculated on the *tall structure objects* class so the false positives (moving objects) did not interfere with this feature.

The proposed Pre+NDT registration approach outperformed the Raw NDT registration in all cases, both in processing speed and crispness. Regarding the *fast movement* dataset the improvement in crispness was remarkable: in some of the corresponding scenes the Raw NDT matching failed completely, matching either the concentric circles on the ground or yielding errors of several meters. In terms of processing time, the proposed method outperformed the Raw NDT registration by an order of magnitude. Also, the proposed workflow is robust enough to perform

Dataset	Number of points (NP)	Crispness by prop. Pre+NDT method	Crisp. by NDT on raw data	Prop. comp speed (sec)	Raw comp speed (sec)
Fast move	434K	36139	64189	13.3	199
Slow move	267K	28635	33160	4.65	48.5

Table 3 Speed and crispness comparison (lower values mean better registration.



(a) Cendroid point track of a moving (top) and parking (bottom) car

(b) Blobs of walking (orange) and standing (blue) pedestrians in the merged cloud.

Fig. 18 Separating moving a static objects in the point cloud

Table 4 Object level evaluation (NP: Number of Points, further notations are given in Sec. 4.2)

DataSet	NO	Obj. Errors		NP	F-rate %	
		FO	MO		Static	Moving
Scene#1	13	3	0	580K	92	89
Scene#2	16	0	0	775K	90	91

well in challenging, noisy real life environments. Our algorithm has been tested on more than 3000 scans, including several different types of scenes (such as avenues, narrow streets, hillside streets, squares, bridges, etc.).

For the proposed object level analysis method (Sec. 4.1.4) we have done quantitative evaluation in two complex scenes. These locations were selected from the aforementioned scenes in a way, that they contain in aggregate 29 objects from four object classes: (i) parking cars, (ii) moving cars, (iii) standing people and (iv) walking pedestrians (see Table 4).

For accurate Ground Truth (GT) generation, we projected the detection result onto the ground, and manually draw GT rectangles around each object in the imaged ground plane. We performed quantitative evaluation both at object and pixel level. At object level, we counted the Number of real Objects (NO), False Objects (FO) and Missing Objects (MO), where we also counted as error if a moving vehicle was classified as a static car etc. At pixel level, we compared the objects silhouette mask to the GT mask, and calculated the F-rate (harmonic mean of precision and recall) of the match. Results in Table 4 report notably accuracy regarding the test sets.

5 Conclusion

In this chapter, we have introduced three different applications of a high speed rotating multi-beam (RMB) Lidar camera. We have demonstrated that the selected sensor provides a flexible tool for various outdoor robot vision problems due to its high temporal refreshing rate and the large field of view. However, the strongly in-

homogeneous density characteristic of the measurements causes different types of challenges in detection and reconstruction tasks. We have proposed novel technical solutions in each application field to overcome the difficulties and we validated the introduced model elements using real world test data sequences.

In the *first* part, we have introduced a novel 3D surveillance framework for detecting and tracking multiple moving pedestrians in point clouds, with focusing on specific challenges raised by the selected RMB Lidar sensor. We have proposed an efficient foreground segmentation model followed by a multi-target tracking module with on-line person re-identification functions, where biometric features were derived from the range and intensity channels of the Lidar data flow. The tracker module was also tested in real outdoor scenarios, with multiple occlusions an several re-appearing people during the observation period. The experiments confirmed, that an efficient 3D video surveillance system can be based on a single RMB-Lidar sensor, whose installation is significantly easier than setting up a calibrated multi-camera system.

In the *second* application area, we have proposed a real time vehicle detection method for autonomous cars equipped with the RMB Lidar sensor. Based on our novel 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.

The *third* domain of interest has been offline dynamic urban scene analysis. We have proposed a simple, yet useful pre-segmentation step for the RMB Lidar measurement frames, which had a great positive effect on the point cloud registration step. For this type of registered data, both high level object detection and scene interpretation methods have been implemented, and the suitability of the approach for virtual city reconstruction has also been demonstrated.

Acknowledgment

This work is partially connected to the i4D project funded by the internal R&D grant of MTA SZTAKI, and it was partially supported 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).

References

Benedek C (2014) 3D people surveillance on range data sequences of a rotating Lidar. Pattern Recognition Letters 50:149–158, special Issue on Depth Image Analysis

Börcs A, Nagy B, Baticz M, Benedek C (2014a) A model-based approach for fast vehicle detection in continuously streamed urban LIDAR point clouds. In: Workshop on Scene Understanding for Autonomous Systems at ACCV, Lecture Notes in Computer Science, Singapore

Börcs A, Nagy B, Benedek C (2014b) 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

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

Douillard B, Quadros A, Morton P, Underwood J, De Deuge M, Hugosson S, Hallstrom M, Bailey T (2012) Scan segments matching for pairwise 3D alignment. In: IEEE International Conference on Robotics and Automation (ICRA), St. Paul, MN, USA, pp 3033 –3040, DOI 10.1109/ICRA.2012.6224788

Geiger A, Lenz P, Urtasun R (2012) Are we ready for autonomous driving? the kitti vision benchmark suite. In: Conference on Computer Vision and Pattern Recognition (CVPR)

Himmelsbach M, Müller A, Luettel T, Wuensche HJ (2008) LIDAR-based 3D Object Perception. In: Proceedings of 1st International Workshop on Cognition for Technical Systems, Munich

Józsa O, Börcs A, Benedek C (2013) Towards 4D virtual city reconstruction from Lidar point cloud sequences. In: ISPRS Workshop on 3D Virtual City Modeling, ISPRS Annals Photogram. Rem. Sens. and Spat. Inf. Sci., vol II-3/W1, Regina, Canada, pp 15–20

Kazhdan M, Bolitho M, Hoppe H (2006) Poisson surface reconstruction. In: Eurographics Symposium on Geometry processing, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp 61–70, URL http://dl.acm.org/citation.cfm?id=1281957.1281965

Kovács L, Kovács A, Utasi A, Szirányi T (2012) Flying target detection and recognition by feature fusion. SPIE Optical Engineering 51(11)

Kuhn HW (1955) The Hungarian method for the assignment problem. Naval Research Logistic Quarterly 2:83–97

Levinson J, Montemerlo M, Thrun S (2007) Map-based precision vehicle localization in urban environments. In: Proceedings of Robotics: Science and Systems, Atlanta, GA, USA

Magnusson M (2009) The three-dimensional normal-distributions transform – an efficient representation for registration, surface analysis, and loop detection. PhD thesis, Örebro University

McNaughton M, Urmson C, Dolan J, Lee JW (2011) Motion planning for autonomous driving with a conformal spatiotemporal lattice. In: International Conference on Robotics and Automation (ICRA), Shanghai, China, pp 4889–4895

Quadros AJ, Underwood JP, Douillard B (2012) An occlusion-aware feature for range images. In: IEEE International Conference on Robotics and Automation (ICRA), St. Paul, MN, USA, pp 4428–4435

Rusu RB, Cousins S (2011) 3D is here: Point Cloud Library (PCL). In: IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China

Schiller I, Koch R (2011) Improved video segmentation by adaptive combination of depth keying and Mixture-of-Gaussians. In: Proc. Scandinavian Conference on Image Analysis, Ystad, Sweden, LNCS, vol 6688, pp 59–68, URL http://dl.acm.org/citation.cfm?id=2009594.2009602

Stauffer C, Grimson W (2000) Learning patterns of activity using real-time tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 22:747–757

Teichman A, Levinson J, Thrun S (2011) Towards 3D object recognition via classification of arbitrary object tracks. In: IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, pp 4034 – 4041

Xiong X, Munoz D, Bagnell J, Hebert M (2011) 3-D scene analysis via sequenced predictions over points and regions. In: IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, pp 2609–2616, DOI 10.1109/ICRA.2011.5980125

Zhang D, Lu G (2002) A comparative study of fourier descriptors for shape representation and retrieval. In: Asian Conference on Computer Vision (ACCV), pp 646–651

Zhang Z (1994) Iterative point matching for registration of free-form curves and surfaces. International Journal of Computer Vision 13(2):119–152