ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume II-3/W1, 2013
VCM 2013 - The ISPRS Workshop on 3D Virtual City Modeling, 28 May 2013, Regina, Canada

TOWARDS 4D VIRTUAL CITY RECONSTRUCTION FROM LIDAR POINT CLOUD SEQUENCES

Oszkár Józsa, Attila Börcs and Csaba Benedek

Distributed Events Analysis Research Laboratory, Computer and Automation Research Institute
H-1111 Budapest, Kende u. 13-17, Hungary
E-mail: firstname.lastname@sztaki.mta.hu

KEY WORDS: Lidar, point cloud registration, scene analysis, reconstruction, moving object detection

ABSTRACT:

In this paper we propose a joint approach on virtual city reconstruction and dynamic scene analysis based on point cloud sequences of a *single* car-mounted Rotating Multi-Beam (RMB) Lidar sensor. The aim of the addressed work 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 for geometric approximation of ground surfaces and building facades due to the sparseness and inhomogeneous density of the individual point cloud scans. In our approach we propose an automatic registration method of the consecutive scans without any additional sensor information such as IMU, and introduce a process for simultaneously extracting reconstructed surfaces, motion information and objects from the registered dense point cloud completed with point time stamp information.

1 INTRODUCTION

Vision based 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 challenges. On one hand, moving object detection, classification, 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. Most existing approaches extract key features first, such as characteristic points, edges, blob centroids, trajectories or histograms, and the recognition process works in a feature space with significantly reduced dimension (Lai and Fox, 2010) compared to the original data. 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. SICK Lidar systems are able to provide dense and accurate point clouds from the environment with homogeneous scanning of the surfaces and a nearly linear increase of points as a function of the distance (Behley et al., 2012). However, since the measurement recording frequency is typically less then 1Hz (often significantly less), these sensors are not well suited to dynamic event analysis.

As an example for alternative solutions of Time-of-Flight (ToF) technologies, (Kim et al., 2012) introduced a portable stereo system for capturing and 3D reconstruction of dynamic outdoor scenes. However in this case, the observed scenario should be surrounded by several (8-9) calibrated cameras beforehand, which fact does not allow quick data acquisition over large urban areas. In addition, the reconstruction process is extremely computation-intensive, dealing with a short 10sec sketch takes several hours, and full automation is difficult due to usual stereo artifacts such as featureless regions and occlusions.

In this paper, we jointly focus on understanding and reconstruction of dense dynamic urban scenes using a single Rotating Multi-Beam (RMB) Lidar sensor (Velodyne HDL-64E), which is mounted on the top of a moving car. Velodyne's RMB Lidar system is able to provide a stream of full 360° point cloud scans with a frame-rate of 20 Hz, yielding that we can capture the scene from view points at about every 30-60 centimeters of distance as the car travels with typical urban traffic speed. Due to its scanning frequency, this configuration is highly appropriate for analyzing moving objects in the scene. 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 (Benedek et al., 2012).

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 rather than complex situation interpretation, and scene visualization, which are addressed in our current work. (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 challenge. (Quadros et al., 2012) presented a feature called *the line image* to support object classification that outperforms the widely used NARF descrip-

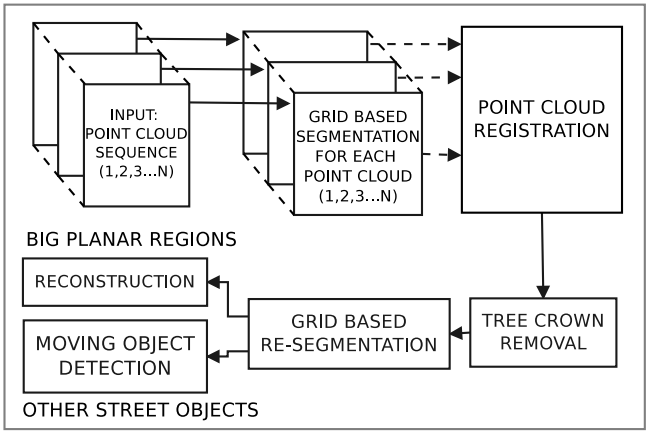

Figure 1: The workflow of the proposed algorithm

tor but requires a computationally expensive principal component analysis (PCA) calculation.

While the above mentioned reference techniques deal mainly with efficient scene interpretation part, (Douillard et al., 2012) proposed a method for registration of the sparse RMB Lidar scans. However, the main issue they address is making the frame matching step robust, as the distance between scans increases. Although this is a significant problem for autonomous driving applications where some of the consecutive frames may be dropped to ensure the real time response of the system, in our scenario we observed having more registration problems due to inhomogeneity of the point clouds and the presence of several moving objects in a crowded street. (Makadia et al., 2006) constructed an automatic but computationally expensive point cloud registration method using large histograms calculated from the surface normal fields, while (Xiao et al., 2012) used a planar-fitting method which works best on flat regions. Recently, (Shen et al., 2011) showed an efficient method for facade reconstruction which uses an iterative adaptive partitioning yielding a flexible and hierarchical representation of the surfaces. However, the previous two models have been tested on more homogeneous scans than our considered RMB Lidar data. The mobile mapping system of (Wang et al., 2012) uses a RMB Lidar within a multi sensor configuration, where point cloud registration is supported by a GPS and an IMU.

The key feature of our approach is that we simultaneously deal with the recognition and mapping issues without relying on any additional sensor information apart from the RMB Lidar stream. Details of the proposed method are introduced in Sec. 2, and evaluation on real data is provided in Sec. 3.

2 PROPOSED POINT CLOUD PROCESSING SYSTEM

The proposed method consists of six main steps, as shown in Fig. 1. *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 floodfill 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.

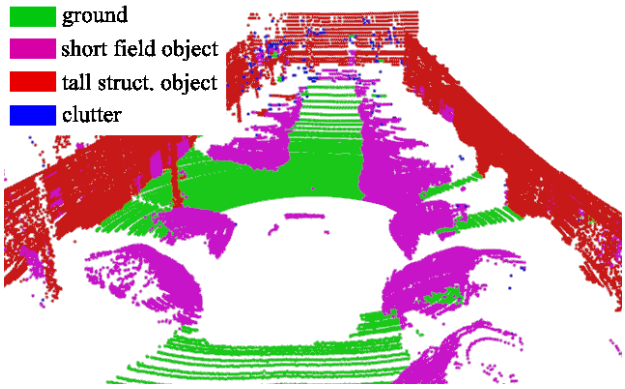

Figure 2: Segmented frame of the Velodyne point cloud stream. Note: figures of this paper are best viewed in color print.

2.1 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.

2.1.1 Cell map segmentation In our system, point cloud segmentation is achieved by a grid based approach. First, we fit a regular 2D grid S with W_S rectangle width onto the $P_{z=0}$ plane, where s denotes a single cell. We used a W_S value between 50cm and 80cm. Smaller grid size is not viable due to the resolution; smaller cells would not have enough points in them to calculate good enough statistical information. On the other hand, larger cell size can result in larger number of falsely classified points, since within a large cell, multiple objects can occur. Near-the-center grid cells have hundreds of points in them, while the point density rapidly decreases as a function of the distance from the sensor. 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. $y_{\max}(s)$, $y_{\min}(s)$ and $\hat{y}(s)$ are the maximum, minimum and average of the elevation values within $\mathcal{P}_s$.

First, local point cloud density is calculated for each cell to extract points of the *clutter* class. From these cells, encapsulating only a few or no points, we can only obtain very limited and probably misleading information regarding the scene structure and the objects, therefore we will neglect these regions in the later model phases. The exact density threshold 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 low. 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 thresh-

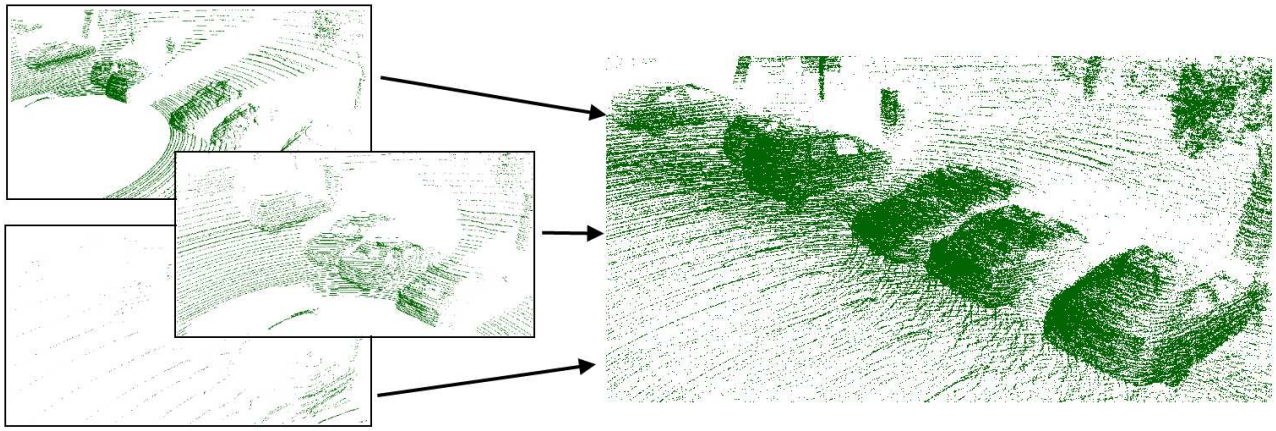

Figure 3: Image sequence showing the registration process. Right side image contains 20 registered scans

olding:

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

where we used $\tau_{\rm gr}$ =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^ν the $\nu \times \nu$ neighborhood of s, and $\gamma_G(s) = \sum_{r \in N_s^\nu} \mathbf{1}_G(s)$, we can obtain a terrain model of scene:

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

where $y_{\rm gr}(s)$ is the estimated ground-elevation value at cell s. The $y_{\rm gr}(s)$ 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 ($\nu = 17$ for cell maps with a size around 400×300). Finally a cell is classified as ground cell iff $\mathbf{1}_G(s) = 1$ and $\hat{y}(s) - y_{\rm gr}(s) < 20$ cm.

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):

$$y_{\max}(s) > 140 \ \vee \ y_{\max}(s) - y_{\min}(s) > 310 \qquad (1)$$

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.

2.1.2 Point cloud labeling 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.

2.2 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 strongly inhomogeneous density of the Lidar clouds, even the static ground points mislead the registration process. The above algorithms often match the concentric circles of the ground (see Fig. 2), which yields that the best match erroneously corresponds to a near zero displacement between two consecutive frames. 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.

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 horizontal configuration and with tilted configurations of the Lidar sensor when mounted atop of

(a) Palace with 48m tall twin towers: result of automatic merging of 30 point cloud frames captured with the tilted configuration.

(b) A triangulation result of the facade of the Great Market Hall

Figure 4: Sample results on facade approximation based on RMB Lidar data with the proposed approach

a vehicle. The horizontal 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. 4.

2.3 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 feature for each point based on the distance and irregularity of its 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. 5). Thereafter, we also refine the separation of *ground*, *tall* and *short street objects* in the registered cloud, using the classification steps introduced in Sec. 2.1.

2.4 Object separation with spatio-temporal floodfill

After removing ground and clutter points from the merged and segmented point cloud, the different objects and surface components are separated with floodfill 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 floodfill steps for

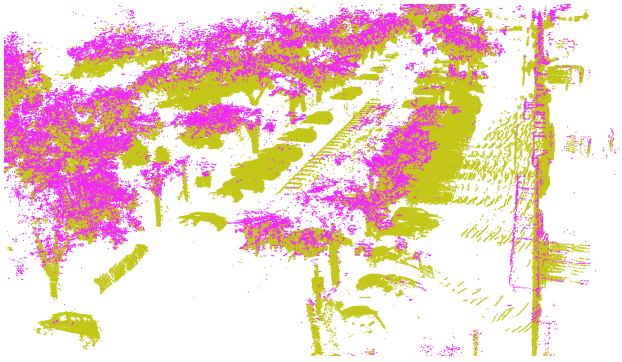

Figure 5: Tree crown detection (marked with purple).

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. 6-8), different object blobs may erroneously be connected due to motion. For this reason, when we create the connected components with the floodfill 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.

2.5 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. 2.2, 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. 4(a) shows a large registered point cloud of a building facade and Fig. 4(b) displays a triangulated mesh obtained with the Poisson surface reconstruction algorithm (Kazhdan et al., 2006).

2.6 Object level analysis

As mentioned in Sec. 2.4, 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 region-level local features proved to be inefficient for motion separation. Instead, we utilize blob-level features: after we extracted the connected blobs of the *short street objects* regions in the merged cloud with floodfill propagation (Sec. 2.4), 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. 6), we can cluster the moving and static object regions as shown in Fig. 7. Fig. 8 demonstrates that with using object size assumptions we can distinguish moving and static pedestrians, respectively vehicles in larger scenes.

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. 3).

3 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.

3.1 Presegmentation and registration

As a quantitative evaluation metrics for the proposed registration algorithm (Sec. 2.2), 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 1. shows the evaluation results of six scenes 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. In all six cases, 10 consecutive point clouds have been registered against each other. 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 six cases, both in processing speed and crispness. In scenes *Slow movement, wide street*, *Fast movement, wide street* and *Slow movement, large square* the Raw NDT registration resulted in seemingly good matches but the crispness values show that even when the full cloud registration succeeds, the Pre+NDT version is more accurate. In the remaining three scenes, the Raw NDT matching failed completely, matching either the concentric circles on the ground or yielding errors of several meters. Denoting by $\widehat{C}_{\text{Prop.}}$ and $\widehat{C}_{\text{Raw}}$ the average crispness values measured with the two methods, and taking the ratio of the difference and the average of the quality factors:

$$\frac{\widehat{C}_{\text{Prop.}} - \widehat{C}_{\text{Raw}}}{\left(\widehat{C}_{\text{Prop.}} + \widehat{C}_{\text{Raw}}\right)/2}$$

we get that with the proposed Pre+NDT technique, the registration is overall 33.35% more accurate.

In terms of processing time, our proposed method outperformed the Raw NDT registration by an order of magnitude. Using the proposed method (once the data is in the memory), the segmentation step runs in real time on a standard CPU.[1] Besides the presegmentation step being fast, it significantly fastens up the registration step also, since a significant amount of noise and interfering moving points are removed. As the registration algorithm tries to match up points, the overall error will be smaller thanks to the "clean" point cloud and the NDT algorithm will converge faster towards the termination criteria. Registering two point clouds containing only stable, stationary points typically takes less than a second while registering two full point clouds typically takes 4-20 seconds (these values depend on the scanning rotation speed, on the number of moving objects and also on the traveling speed of the sensor platform).

[1] The test environment has an Intel Core i7 processor and 8 gigabytes of DDR3 RAM

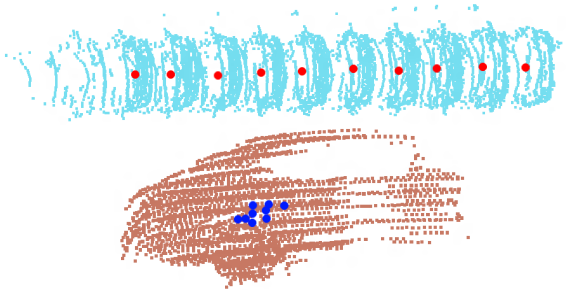

Figure 6: Representation of cendroid points distribution as a feature. On the top the centroids follow big elongated line in case of moving vehicle, while on the bottom the centroids movements stay in a certain region.

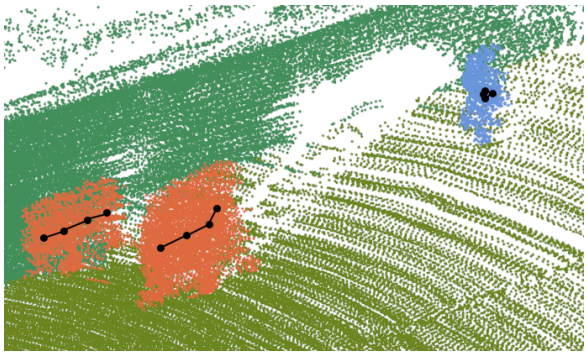

Figure 7: Blobs of two moving pedestrians (orange) and a standing person (blue) on the merged and segmented point cloud. Trajectories of object centroids in the separate time frames are marked with black.

Also, the proposed workflow is robust enough to perform 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.).

3.2 Object detection

For the proposed object level analysis method (Sec. 2.4) we have done quantitative evaluation in two complex scenes. These data sets were selected from the aforementioned scenes in a way, that they contain several objects from the classes parking cars, moving cars, standing and walking pedestrians. The two scenes contain 29 street objects (see Table 2) in aggregate.

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

Dataset description	Number of points (NP)	Crispness by prop. Pre+NDT method	Crispness by NDT on raw data	Speed with the prop. preseg-mentation (sec)	Speed without the preseg-mentation (sec)
Fast movement, small square	237K	42082	88601*	20.00	331.29*
Slow movement, wide street	175K	35829	39892	6.46	53.82
Fast movement, wide street	116K	32650	34952	3.55	42.98
Fast movement, wide street 2	81K	23747	34634*	7.73	38.39*
Slow movement, large square	56K	16296	16565	1.40	35.81
Very slow movement, large square	36K	12860	26256*	0.90	42.07*

Table 1: Speed and crispness comparison (lower values mean better registration. * denotes failed registration on the full point cloud)

DataSet	NO	Obj. Errors		NP	F-rate %	
		FO	MO		Static	Moving
Set#1	13	3	0	580K	92	89
Set#2	16	0	0	775K	90	91
Overall	29	3	0	1.4M	91	90

Table 2: Numerical evaluation of the detection results obtained by the proposed object level analysis. Number of real Objects (NO), Missing Objects (MO), False Objects (FO), Number of Points (NP) and pixel level F-rates (in %) are listed for each data set, also and in aggregate.

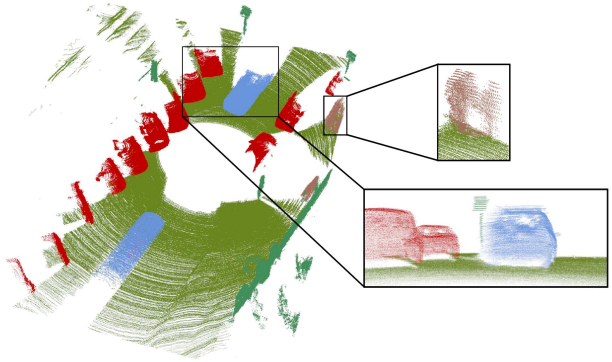

Figure 8: Demonstration of detecting different moving and static object regions in the point cloud. Color codes: dark green - static objects, light green - ground, red - static vehicles, blue - moving vehicles, light brown - moving pedestrian

recall) of the match. Results in Table 2 report notably accuarcy regarding the test sets.

4 CONCLUSION AND FURTHER WORK

The results above show the success of the simple, yet useful pre-segmentation step that has a great positive effect on the point cloud registration. Dividing the initial point cloud into multiple semantic classes and using only the relevant points results in a faster, more stable and more accurate registration. On this registered data, both high level object detection and scene interpretation is possible and it is suitable for virtual city reconstruction also.

Future work will focus on higher level scene interpretation such as dividing the road class into smaller semantic units e.g.: sidewalks, lanes, parking lots, road intersections. Also, additional semantic classes can be extracted from the scenes such as traffic signs, traffic lamps, crosswalks, etc.

Also, future work will be done on fusing 3D and RGB data. Having RGB data that is registered to the point cloud, the reconstructed city models can be textured, obtaining a higher level of visual experience.

This work is connected to the i4D project funded by the internal R&D grant of MTA SZTAKI. Csaba Benedek also acknowledges the support of the János Bolyai Research Scholarship of the Hungarian Academy of Sciences and the Grant #101598 of the Hungarian Research Fund (OTKA).

REFERENCES

Behley, J., Steinhage, V. and Cremers, A., 2012. Performance of histogram descriptors for the classification of 3D laser range data in urban environments. In: IEEE International Conference on Robotics and Automation (ICRA), St. Paul, MN, USA, pp. 4391–4398.

Benedek, C., Molnár, D. and Szirányi, T., 2012. A dynamic MRF model for foreground detection on range data sequences of rotating multi-beam Lidar. In: International Workshop on Depth Image Analysis, LNCS, Tsukuba City, Japan.

Douillard, B., Quadros, A., Morton, P., Underwood, J., De Deuge, M., Hugosson, S., Hallstrom, M. and 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.

Douillard, B., Underwood, J., Kuntz, N., Vlaskine, V., Quadros, A., Morton, P. and 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.

Kazhdan, M., Bolitho, M. and Hoppe, H., 2006. Poisson surface reconstruction. In: Eurographics Symposium on Geometry processing, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp. 61–70.

Kim, H., Guillemaut, J.-Y., Takai, T., Sarim, M. and Hilton, A., 2012. Outdoor dynamic 3-D scene reconstruction. IEEE Trans. on Circuits and Systems for Video Technology 22(11), pp. 1611 –1622.

Lai, K. and Fox, D., 2010. Object recognition in 3D point clouds using web data and domain adaptation. International Journal of Robotics Research 29(8), pp. 1019–1037.

Magnusson, M., 2009. The Three-Dimensional Normal-Distributions Transform – an Efficient Representation for Registration, Surface Analysis, and Loop Detection. PhD thesis, Örebro University.

Makadia, A., Patterson, A. and Daniilidis, K., 2006. Fully automatic registration of 3D point clouds. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1, pp. 1297 – 1304.

Quadros, A. J., Underwood, J. P. and 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.

Shen, C. H., Huang, S. S., Fu, H. and Hu, S. M., 2011. Adaptive partitioning of urban facades. In: SIGGRAPH Asia Conference, ACM, New York, NY, USA, pp. 184:1–184:10.

Teichman, A., Levinson, J. and 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.

Wang, R., Bach, J., Macfarlane, J. and Ferrie, F., 2012. A new upsampling method for mobile LiDAR data. In: IEEE Workshop on Applications of Computer Vision (WACV), pp. 17–24.

Xiao, J., Adler, B. and Zhang, H., 2012. 3d point cloud registration based on planar surfaces. In: Multisensor Fusion and Integration for Intelligent Systems (MFI), 2012 IEEE Conference on, IEEE, pp. 40–45.

Xiong, X., Munoz, D., Bagnell, J. and 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.

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