Advanced Methods for Environment Analysis Based on Lidar Data

A thesis submitted for the degree of
Doctor of Philosophy

Örkény Ádám H. Zováthi

Scientific advisor:
Csaba Benedek, D.Sc.

Faculty of Information Technology and Bionics
Pázmány Péter Catholic University

Budapest, 2023

I would like to dedicate this thesis to my family

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Prof. Csaba Benedek for his continuous support, motivation and patience during my Ph.D. study and work. Special thanks to Prof. Tamás Szirányi, head of the Machine Perception Research Laboratory for supporting my research at the Institute for Computer Science and Control (SZTAKI). I thank my colleagues for their works and advices: Csaba Benedek, the leader of our research group, Zsolt Jankó, the company advisor of my KDP research, furthermore Balázs Nagy, Lóránt Kovács, Balázs Pálffy, József Kövendi, Yahya Ibrahim and László Tizedes. Pázmány Péter Catholic University (PPCU) is also gratefully acknowledged, thanks to Prof. Gábor Szederkényi for the opportunity to study there. Special thanks to those whom I may have not mentioned by name but who supported me directly or indirectly in accomplishing my Ph.D. research.

I thank the reviewers of my thesis for their work and valuable comments.

For financial support, thanks to the National Research, Development and Innovation Fund, under the ÚNKP-20-3 and ÚNKP 21-3 New National Excellence Program and under the KDP-2020 Cooperative Doctoral Program (KDP-977852). My research at SZTAKI was also supported within the frameworks of the Autonomous Systems National Laboratory and the Artificial Intelligence National Laboratory programs and the TKP2021-NVA-01, OTKA K-120233, and K-143274 projects.

Last but not least, I would like to thank my family for supporting me throughout these Ph.D. years and my life in general. My mother who raised me with her unconditional love that I will never be able to pay back even if I live for a thousand years, my father who always motivated me to dream big and taught me that hard work always pays off, my brothers, Bendi, Kende, Csani and Domi and my beautiful wife Kata who motivated and supported me even in the hardest times.

To my grandparents for their neverending blessings and to my godmother for her continuous support.

Abstract

In this dissertation, I introduce novel methods for real-time environment analysis using point cloud measurements of different Lidar (Light detection and ranging) sensors, which can benefit both mobile robotics and autonomous driving. The thesis consists of two main parts. In the first part, I exploit dense spatial information of high-density 3D city maps to improve the vehicles' onboard Lidar-based perception capabilities. I present (i) a cross-source point cloud registration approach for accurate self-localization, and an extension of this method for robust pose tracking, (ii) a cross-source change detection algorithm between registered point clouds for low-level scene segmentation, and (iii) a high-level application of the above methods for improved dynamic object detection. In the second part, I introduce a novel method for enhancing the quality of sparse and noisy Lidar depth measurement sequences, without relying on any external data or information sources. The proposed solutions contain traditional geometry-based algorithms and deep learning-based methods. This hybrid approach aims to incorporate both geometric understanding and learned representations of the Lidar measurements. All proposed methods have been evaluated in real-life urban traffic scenarios and experimentally compared against the state of the art, showing significant advantages.

Contents

Chapter 1

Introduction

1.1 Motivation

In our modern world, the coexistence of humans and intelligent machines already plays a significant role and the tendency is likely to continue and even evolve in the future. Software and hardware technology have significantly advanced, enabling machines to perform various tasks and assist humans in several domains. During our daily life, we, humans interact with the world in every minute. We continuously perceive our environment using various senses: mostly our vision system, but also by hearing, touch, taste or smell. These senses allow us to interpret the world around us, gathering information about objects, people and events to understand the overall context of our surroundings. Similarly, intelligent machines such as robots or vehicles need advanced two- (2D) or three-dimensional (3D) sensors to continuously monitor and analyze their environment, facilitating a better understanding of their surroundings and enabling them to make informed decisions and to navigate and operate safely.

Nowadays, the advanced perception systems of modern robots and vehicles [8–10] usually combine multiple sensors such as cameras, radars (Radio detection and ranging), and Lidar (Light detection and ranging) scanners to provide complementary and redundant information. In general, Lidars capture accurate 3D point measurement flows with high acquisition speed [11]. As active laser-based sensors, they efficiently perform under different illumination and lighting conditions, but may provide noisy measurements in adverse weather situations like fog, heavy rain, or snow. Cameras are characterized

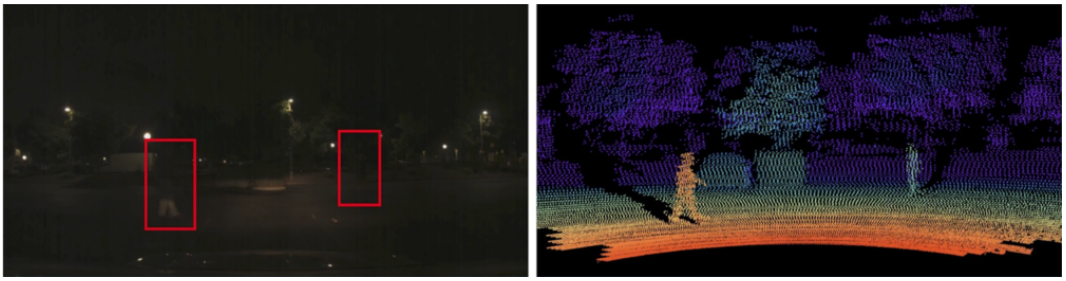


Figure 1.1: Comparison of an automotive camera image and a Lidar point cloud in dark conditions with streetlights and low beam headlights (Source: [103]).

by their ability to capture high-resolution and high-quality image sequences with rich color information, however, they are easily affected by the illumination changes of the scenes [12]. Radars excel at providing accurate velocity and distance measurements in all weather conditions, but they typically have lower spatial resolution compared to Lidars or cameras, therefore they may struggle to provide detailed information about the environment.

Although it is inevitable that the most comprehensive perception capabilities can be achieved through integrating the different sensor modalities [13], there can be situations when Lidars may be the only sensors that are able to robustly provide detailed and accurate measurements, for example in case of a sudden sensor failure or temporal field-of-view (FoV) occlusion, or in scenarios with limited lighting conditions such as darkness during nighttime driving (see Fig. 1.1) or compromised visibility due to direct sunlight when the quality of camera data usually degrades.

1.2 Thesis objectives

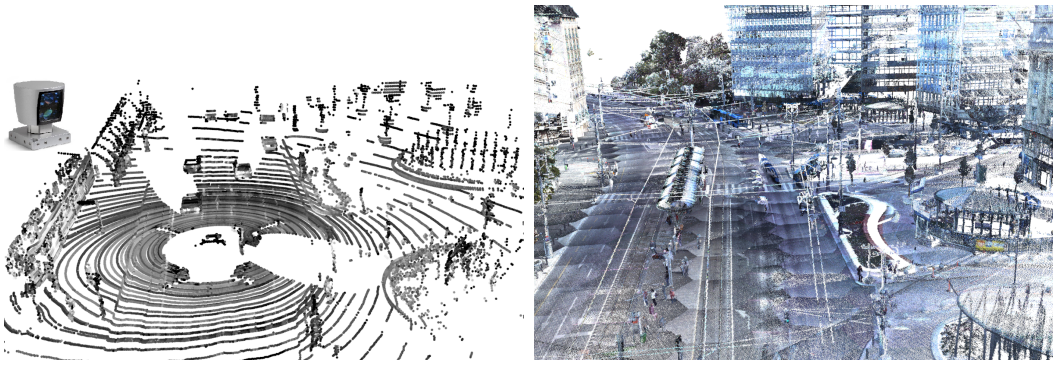
In this thesis, we focus on advanced environment analysis of moving robot or vehicle platforms, using onboard measurements of solely Lidar sensors. As the main challenge, in complex urban environments, the presence of various obstacles, such as street furniture elements (e.g., traffic signs and lamps, bus or tram stations, etc.), vehicles, and pedestrians can lead to occlusions and to the reflection of laser beams in multiple directions, resulting in incomplete or sparse point cloud data. This phenomenon can lead to issues such as Lidar-based algorithms misidentify obstacles or have difficulties in navigating.

We address this challenge in two specific manners. In the first part, we integrate the onboard Lidar measurements with prior, detailed 3D map data to provide a more comprehensive perception of the environment. In the second part, assuming that 3D maps may not be available, we provide an algorithm for the densification of the sparse Lidar measurements, in order to make the raw data more useful for navigation, mapping, and analysis in the future.

1.2.1 Map-Lidar fusion for real-time scene analysis

The first topic focuses on the fusion of different Lidar-based data modalities. A long-term vision of the conducted research work is to facilitate the joint exploitation of the measurements from the vehicles' real-time sensing platforms and offline spatial database content of the newest Geo-Information System (GIS) solutions. On one hand, the proposed new algorithms may help autonomous vehicles (AVs) to obtain relevant spatial 3D map information for decision support in real-time, for example, to enable safer moving vehicle/pedestrian detection or behavior prediction in the future. On the other hand, they also may provide opportunities for extending and updating the GIS databases based on the sensor measurements of the vehicles in everyday traffic.

In the addressed scenario, we assume a vehicle that captures onboard real-time 3D measurements [14] taken by a rotating multi-beam (RMB) Lidar sensor (see Fig. 1.2(a)). Beyond the RMB Lidar sensor, we also assume that the



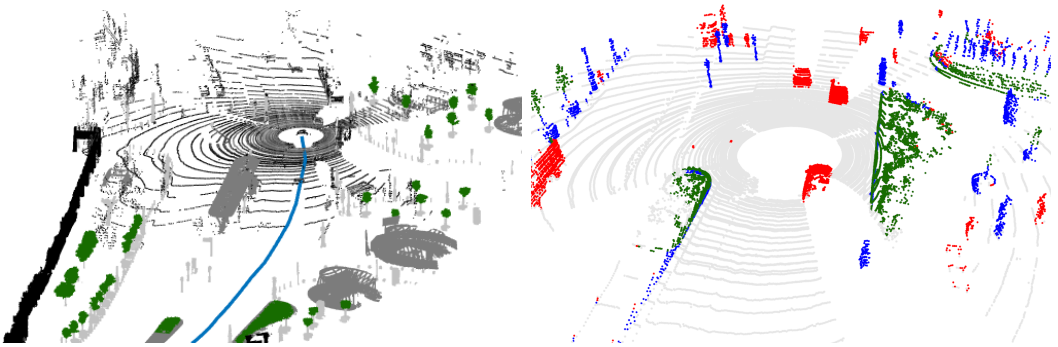
(a) A sparse onboard Lidar measurement frame

(b) Dense point cloud segment recorded by an up-to-date MLS platform

Figure 1.2: Comparison of (a) sparse RMB Lidar-based and (b) dense MLS point clouds captured in the same inner-city area.

vehicle carries a Global Navigation Satellite System (GNSS) receiver, however, we expect that in various dense city regions the global positioning might be inaccurate, providing location errors up to several meters. To augment the AVs' limited RMB measurements with prior beliefs, we obtain detailed segmented 3D environment models of dense urban areas. In industrial practice, vector-based high-definition (HD) maps are often adopted for this purpose [15], as they store precise and high-quality metadata about the static parts of the environment [16]. Following a different approach, we utilize offline integrated Mobile Laser Scanning (MLS) measurements that provide accurate and georeferenced point clouds [17, 18] (see Fig. 1.2(b)). Without any vectorization, as preprocessing we segment the raw and noisy MLS data using an automatic voxel-based point cloud segmentation technique [19] and remove all regions that contain ground areas or dynamic objects, and consider the remaining point cloud segments as highly detailed reference landmarks for the AVs' onboard RMB measurements.

As the main outputs, we aim first to achieve accurate (up to centimeters) global localization and pose tracking (Fig. 1.3(a)) of the AVs. Using the localization results, our next goal is to separate changed dynamic (including traffic participants, urban renewal areas) or seasonally varying (vegetation areas, tree crowns, bushes) regions, and unchanged static (among others street furniture, traffic lights, signs) areas (Fig. 1.3(b)) from the AVs' RMB measurements



(a) Localization and pose tracking (b) Change detection

Figure 1.3: Goals of the first topic: accurate localization and pose tracking by aligning the vehicles's onboard Lidar data to the semantic prior 3D map (a), then detecting changes between the registered point clouds (b). Color codes for the onboard Lidar data (b): static regions are marked by blue, seasonally varying regions by green, and dynamic changes by red.

exploiting the prior information obtained from the segmented MLS data.

In the above context, we propose four significant contributions that improve the state of the art:

1. A novel efficient algorithm is introduced for the *registration* of sparse RMB and dense MLS point clouds with a relatively poor initial alignment with position errors up to 12 meters, and orientation errors up to ± 60 degrees. In practice, this method can compensate for the global positioning error of the AVs in dense urban areas where high-quality GPS signals cannot be received.

2. An efficient and robust algorithm is presented for *tracking* the vehicle movement even in highly occluded temporal scenes, by estimating poses from the registration results and fusing them in a constant velocity model-based position-only-measured Kalman filter.

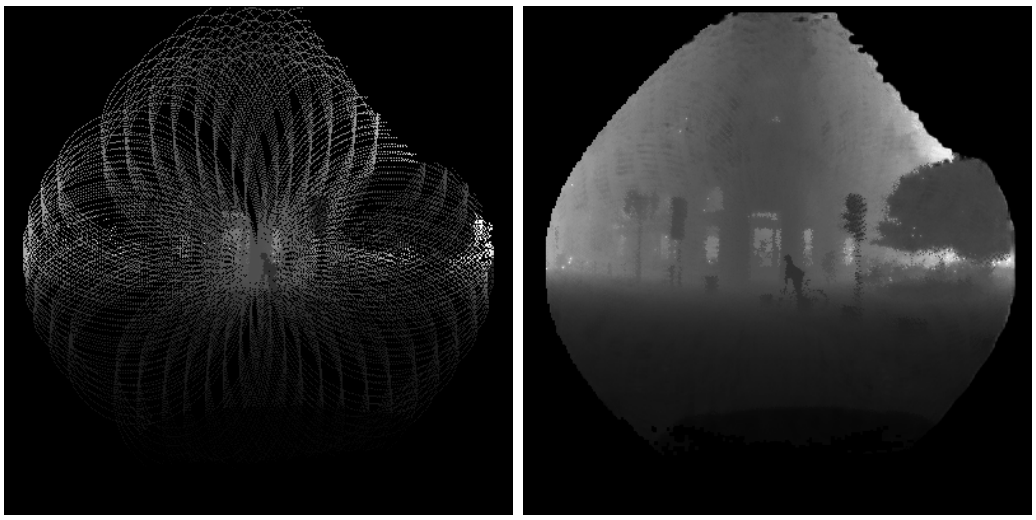
3. A new Markov Random Field-based approach (RangeMRF) is proposed for multi-class *change extraction* and classification (dynamic, seasonal, or no change) between the registered point clouds in a *lossless* way by 2D range image representations.

4. An efficient utilization of the proposed point-level change detection approach is demonstrated for enhancing the performance of state-of-the-art Lidar-only object detection.

To the best of our knowledge, this is the first work that addresses a joint multi-sensory point cloud registration and change detection problem by fusing sparse RMB point clouds and preliminary recorded MLS data, where, as a key challenge, the MLS point clouds are in several regions 100-1000 times denser than the corresponding RMB measurement segments. For the above reason, we constructed a new dataset, called the *SZTAKIBudapest* Benchmark, which contains real RMB Lidar measurements and industrial MLS data, enabling the research of the concerning problem and validation of the proposed cross-source registration, localization, change, and object detection approaches.

1.2.2 Real-time densification of sparse Lidar data

The second topic focuses on the densification of sparse depth measurements of Lidar devices without external sources or information, by exploiting tem-



(a) Example sparse input to our method (b) The output of our proposed method

Figure 1.4: Goal of the second topic: completion of sparse depth measurements captured by a non-rotating circular scanning Lidar.

poral and spatial patterns of the raw sensor data stream. In situations when there is no pre-existing map data available, the vehicles must heavily rely on the sparse real-time Lidar measurements. Without a comprehensive and dense point cloud, they may struggle to perceive the environment accurately and to make informed decisions. In such scenarios, accurate and fast densification of the sparse Lidar measurements could play a critical role.

In the addressed scenario, we assume a robot or vehicle platform that is equipped with a single Lidar that exhibits non-repetitive circular scanning (NRCS) patterns. Here, as main contribution, we propose a novel deep learning technique for the densification of sparse consecutive measurements of NRCS Lidars. The proposed method provides a dense depth data stream with both high spatial resolution and accuracy (see Fig. 1.4) and aims to contribute to more advanced scene understanding or mapping functionalities in the future.

While the main goal of our work is to propose an algorithm that can accurately deal with real Lidar measurement sequences, it is challenging to provide dense, spatially precise depth information from the real world due to the independent movements of dynamic objects of the scene including the ego-motion of the robot or vehicle. To overcome this limitation, we constructed a new urban dataset, that – to our best knowledge as the first open Benchmark in this field – comprises various simulated and real-world NRCS Lidar data

samples, allowing us to simultaneously train and test methods on synthetic data with ground truth (GT), and to validate the result via real NRCS Lidar measurements.

1.3 Structure of the thesis

The structure of the thesis mainly follows the order of the two topics outlined above. In Chapter 2, we introduce the Lidar technology in detail, the main application areas, and the typical sensor devices that were applied in the thesis for the addressed research problems. In Chapter 3, we present the constructed new datasets using the measurements of these devices. In Chapter 4, we present the proposed methods for 3D map-based scene analysis: First a cross-source localization and pose tracking technique, which is followed by the description of the proposed cross-source change detection algorithm and the object detection method. In Chapter 5, we introduce our proposed method for Lidar data densification. Finally, in Chapter 6, we conclude the main achievements of the thesis and discuss further related research problems.

Chapter 2

Lidar technology

2.1 Introduction and short history

The word Lidar is derived from the term *Light detection and ranging*, and refers to a technology that uses electromagnetic waves in the visible light and infrared spectrum. In the history, one of the first attempts to measure distance by light beams was made in the 1930s, when searchlights were used to study the structure of the atmosphere and to determine the heights of clouds. The main development started however only in the early 1960s, thanks to the invention of laser. Lidars started first operating in the visible domain, then in the near infrared (NIR) and thermal infrared (TIR) regions. Many Lidars are now being developed in the eye-safe, short-wave infrared (SWIR) domain, having a wavelength of about 900-1000 nanometers.

In general, today's Lidar sensors work by the principle of emitting laser pulses and measuring the time until their return after reflecting off objects in the environment. The corresponding distance to each emitted pulse is calculated based on the following time-of-flight (ToF) equation:

$$d = \frac{t_\Delta \cdot c}{2}$$

Here d represents the distance to the given object, c is the speed of light propagation and t_Δ is the measured echo time of the emitted laser beam.

In theory, this principle is very similar to the Radar technology – named after *Radio detection and ranging* – that uses radio- or microwaves, just Lidars use much shorter wavelengths. This means in general that Lidars will have much better angular resolution than radars but will not see through fog or

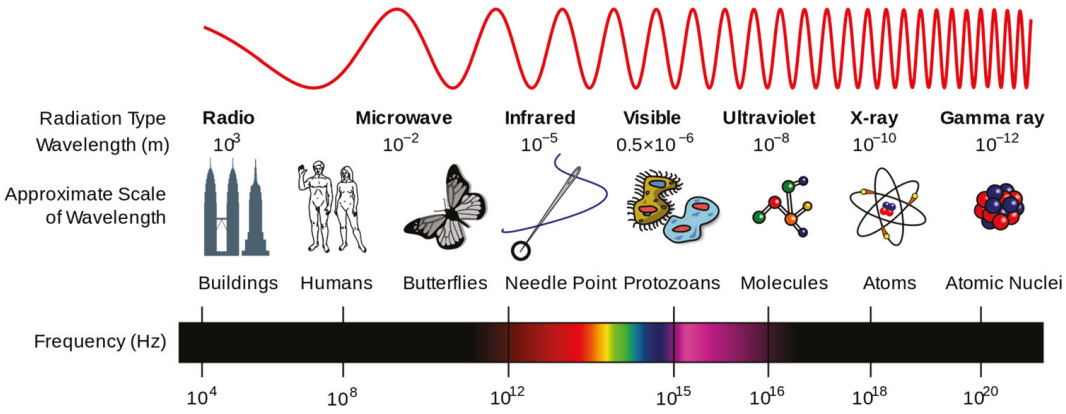


Figure 2.1: Comparison chart of different electromagnetic waves (Source: [20]).

clouds [20]. A comparison chart about the typical wavelengths and frequencies of electromagnetic waves is displayed in Fig. 2.1.

In general, the modern Lidar sensors integrate several laser emitter and receiver pairs, and follow a specific scanning pattern to collect a series of distance measurements that form a 3D point cloud representation of the scene. The intensity of the reflected light and therefore the measured surface's reflectivity can be usually measured as well.

In recent years, the Lidar technology has witnessed remarkable advancements, leading to significant improvements like increased spatial resolution, scanning speed, and accuracy, while also becoming more compact and cost-effective. As a result of this tendency, Lidar devices play a key role today in environment perception and understanding.

2.2 Main application areas and types of Lidar scanners

Lidar maps find use today in various application domains including autonomous robots and vehicles or city management. In the next subsections, I briefly introduce these two main areas, the corresponding Lidar types, and the exact sensor devices that were used during this thesis.

2.2.1 Autonomous driving and mobile robotics

Lidar is a crucial component for autonomous robots and vehicles, helping them navigate and perceive their surroundings. Autonomous vehicles and mobile robots demand *real-time 3D data acquisition* and processing techniques for identifying obstacles, pedestrians, and other vehicles, enabling safe and efficient autonomous driving. In the last decade, repetitive, typically rotating multi-beam (RMB) Lidar sensors [1] have been utilized for this purpose.

2.2.1.1 Rotating multi-beam Lidars

RMB Lidars can produce real-time point cloud streams (300 thousand - 2 million points per second). These sensors include a rotating element that is responsible for spinning the sensor head 360 degrees horizontally, with a contin-

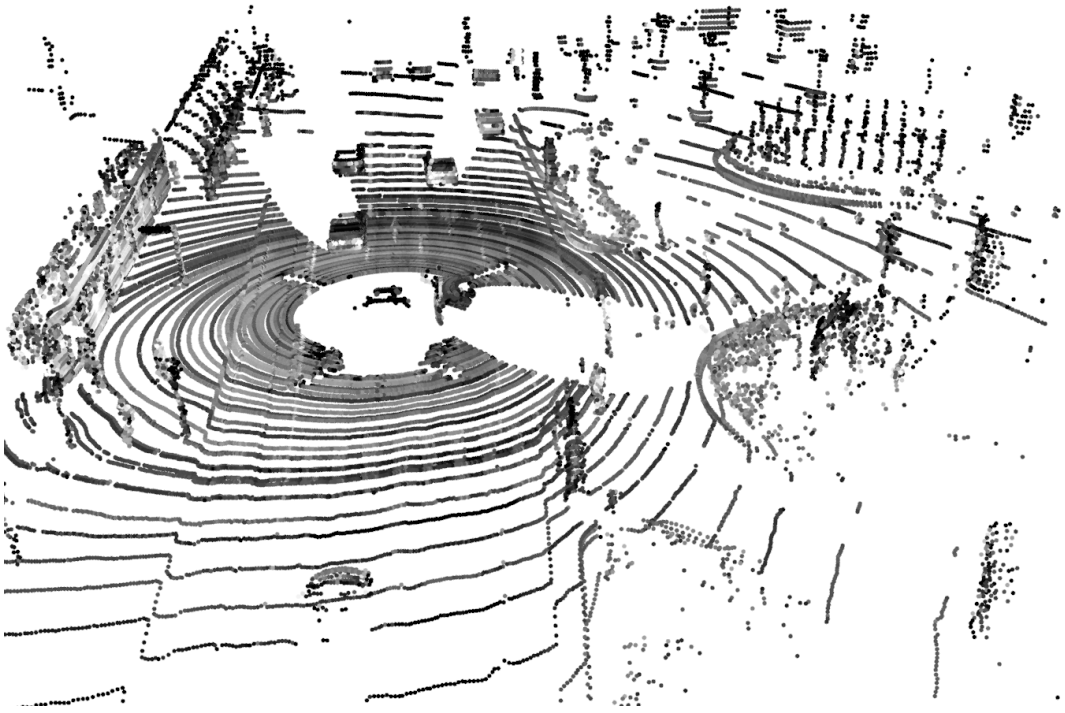


Figure 2.2: Point cloud frame captured by the Velodyne HDL 64E RMB Lidar.

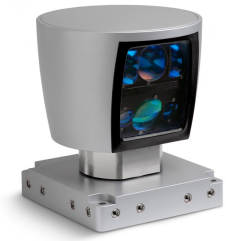


Figure 2.3: The Velodyne HDL 64E Lidar sensor.

uous rotation frequency of 5-20 Hz. Therefore, RMB Lidars can capture data from all directions around the sensor's location. The sensor measurements are typically collected and processed in point cloud frames (see Fig. 2.2), where the term *frame* refers to a single horizontal turnaround of the sensor head. The captured data within a single time frame is notably sparse and non-uniformly distributed: the point clouds have low vertical resolutions, while their densities decline rapidly if the objects are located further from the sensor. Also, the sensors' FoV coverage is constant through the whole scanning process: Their vertical resolution is fixed by the number of laser beams (typically between 16-128), while their horizontal resolution depends on the sensor's rotation frequency (5-20 Hz).

As a popular RMB Lidar, we used the Velodyne HDL 64E (Fig. 2.3(a)) sensor [8, 9] to evaluate the developed algorithms in the thesis. The Velodyne HDL 64E sensor [104] utilizes 64 laser channels with a vertical FoV of 26.9° and delivers a real-time 360° horizontal FoV. The rotation rate is variable from 5 Hz to 20 Hz which enables the user to determine the density of data points generated by the sensor. With a high rotation rate, the sensor can generate point clouds of up to ca. 2.2 million points per second with a sensing range of up to 120 meters. The spatial accuracy is around 1-2 centimeters in the sensor's own coordinate system, but the point density quickly decreases as a function of the distance from the sensor and it shows typical ring patterns. The HDL-64E is designed to operate over a wide temperature range and challenging environments to support diverse operating conditions and applications. The original price of this device was around 75.000 USD. Please note that in 2023, there are more recent RMB Lidars such as the Ouster OS1 [105] with the same resolution and at a more affordable price, costing around 8.000 USD.

2.2.1.2 Non-repetitive circular scanning Lidars

Alternatively to the widespread RMB technology, recent non-repetitive circular scanning (NRCS) Lidar sensors are also capable of providing measurements for real-time scene analysis, at a significantly lower cost compared to RMB Lidars [21], by using single- or multi-line lasers combined with high-speed scanning on a circular path. Unlike RMB Lidars whose FoV coverage is constant through the whole scanning process, NRCS Lidars are able to densely map large areas from a given scanning position due to their special scanning technology which follows non-repetitive patterns (Fig. 2.6). The main challenge is here to efficiently balance between the spatial and the temporal resolution of the recorded range data using a suitable integration window [22].

In this thesis, we used the very recent Livox AVIA (see Fig. 2.5) sensor [106]. The Livox AVIA sensor uses a multi-line laser combined with high-speed scanning. This results in a point cloud data capturing rate of up to 240 thousand

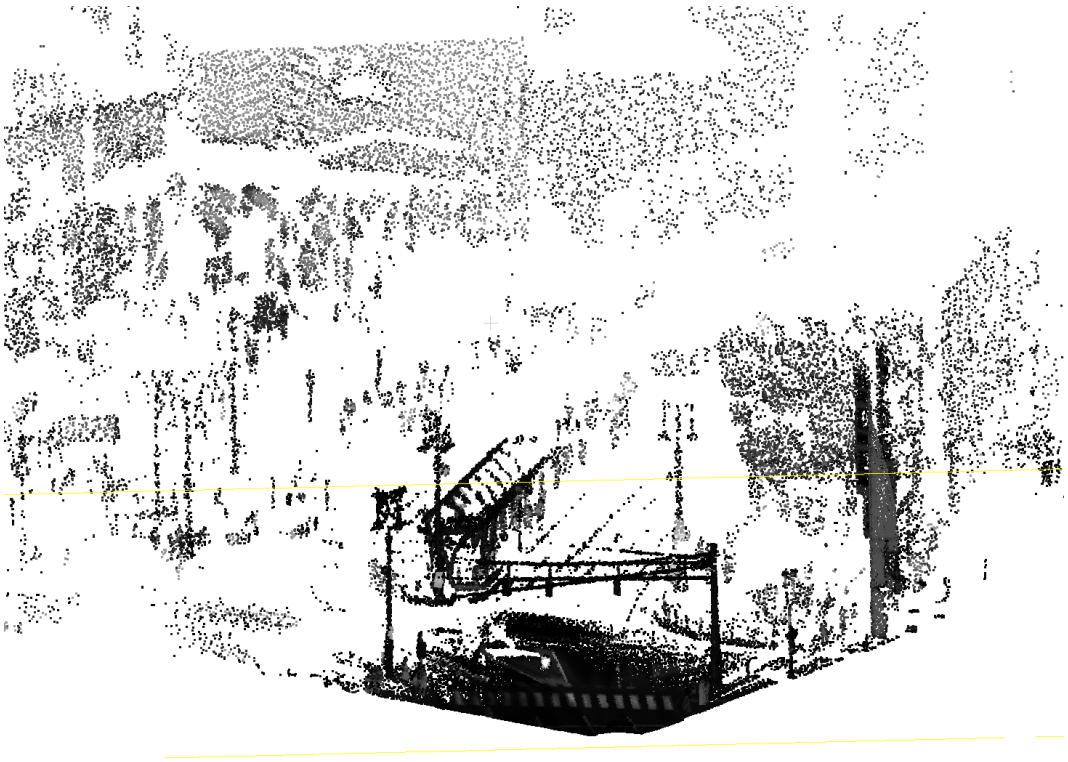


Figure 2.4: Point cloud captured by the Livox AVIA NRCS Lidar sensor within 1000 *ms.*

Figure 2.5: The Livox AVIA NRCS Lidar sensor.

points per second with a detection range of up to 320 meters. The Livox AVIA sensor has six Lidar beams organized in a linear beam array, which is moved and rotated inside the sensor following a non-repetitive rosetta pattern. As a result, the sensor has a FoV of 77.2° vertically and 70.4° horizontally, while the scanning density is higher in the center of the FoV compared to the surrounding area (Fig. 2.4). The Livox AVIA sensor costs 1.500 USD and it is suitable for the majority of use case scenarios in traditional mapping, mobile robotics, and urban (low-speed) autonomous driving.

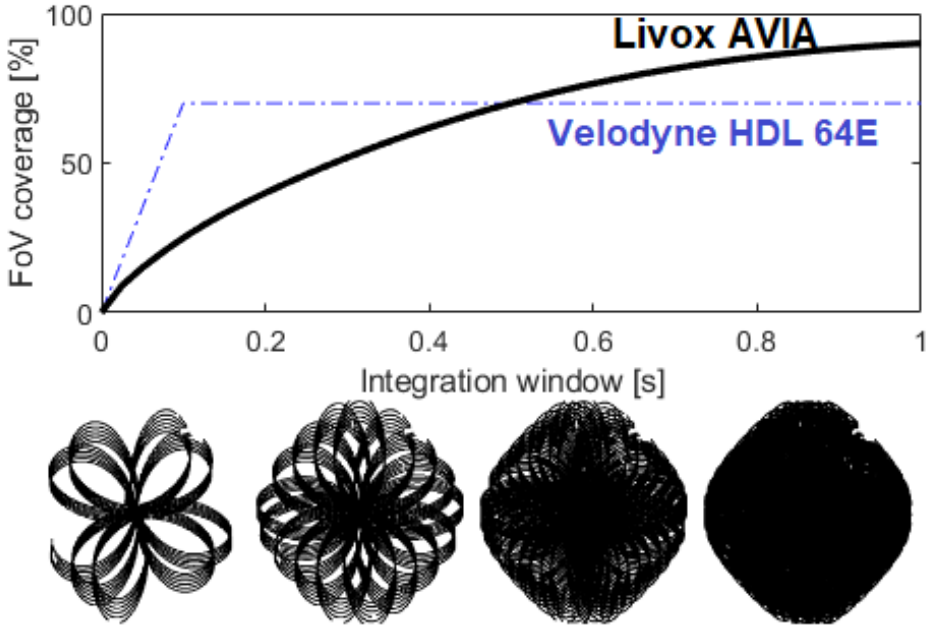


Figure 2.6: Non-repetitive sampling strategy of the Livox AVIA NRCS Lidar. The circular scanning produces typical rosetta patterns which vary across different time frames.

Fig. 2.6 displays a comparison between the field-of-view coverage of a traditional rotating 64-beam Lidar and the Livox AVIA sensor. While using a rotating Lidar, if the integration time is greater than the rotation period, the field-of-view coverage is fixed (marked by blue on Fig. 2.6). Unlike that, using the Livox AVIA sensor, due to its aperiodic sampling strategy, the field-of-view coverage varies based on the integration time from 0 to 100% (marked by black on Fig. 2.6). The lower subplot displays these different field-of-view coverages using integration times of 50 ms, 200 ms, 500 ms, and 1 s, where the covered areas are represented by black.

2.2.2 City management and maintenance

Lidar technology is valuable for urban planning and management as well. Lidar-equipped vehicles or drones can scan and map entire cityscapes quickly

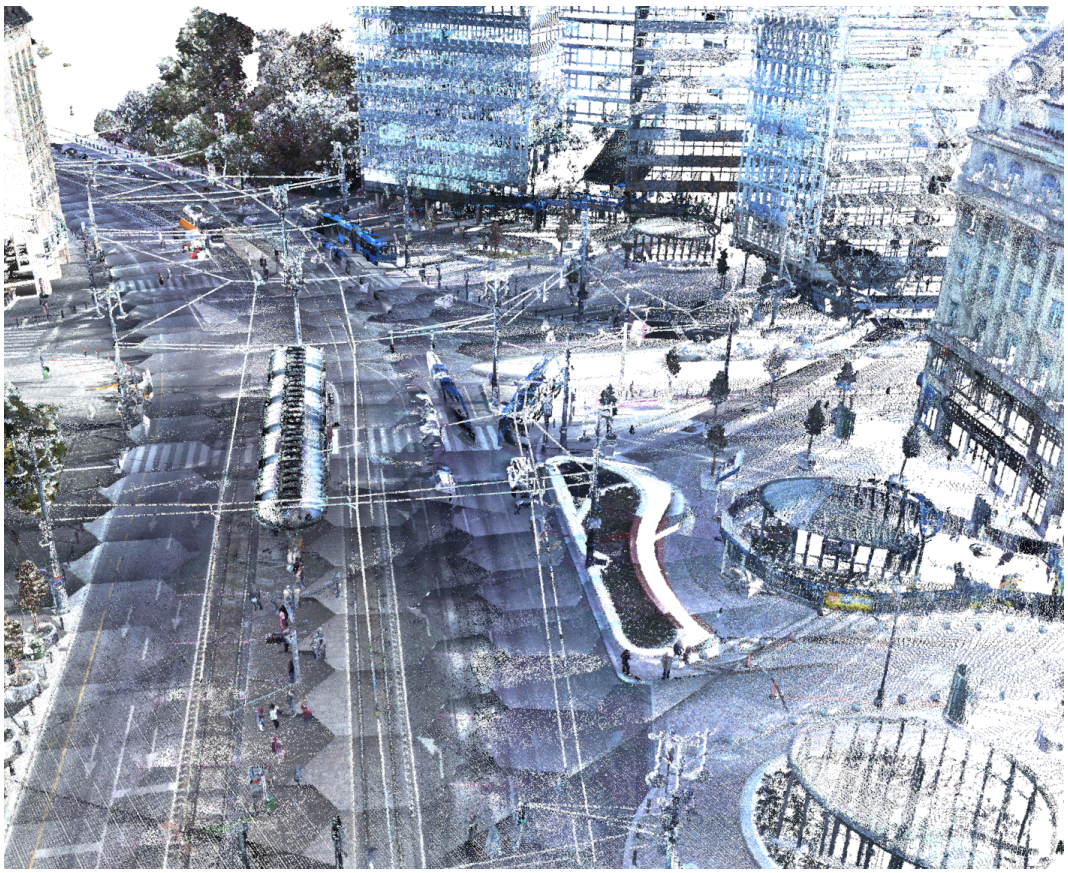


Figure 2.7: Point cloud captured by the Riegl VMX450 MLS system.

Figure 2.8: The Riegl VMX450 MLS system.

and provide very detailed and accurate 3D spatial maps from the environment, which are stored and maintained in new-generation Geo-Information Systems (GIS). These 3D city maps help with various applications, such as assessing infrastructure conditions or monitoring urban development. Recent Terrestrial (TLS) and Mobile Laser Scanning (MLS) platforms equipped with time-synchronized Lidar sensors, calibrated cameras, and navigation units are common choices for such applications, as they provide dense, accurate, and feature-rich point clouds precisely registered to a geo-referenced global coordinate system.

In the thesis, we used the measurements of the Riegl VMX450 MLS system (Fig. 2.8) [107] that offers extremely high measurement rates providing dense, accurate (up to global accuracy of a few centimeters), and feature-rich point cloud data with a quite uniform point distribution even at high driving speeds (see Fig. 2.7). The roof-carrier-mounted measuring head integrates two Riegl VQ-450 laser scanners, a well-designed, calibrated camera platform with up to six digital cameras, inertial measurement units (IMUs), and GNSS equipment, all of them housed under an aerodynamically-shaped protective cover. By the combination of precise laser scanning technology and high-resolution imaging, the VMX450 system enables fast and accurate 3D data acquisition for a variety of applications, including infrastructure management, city mapping, urban planning, and road surveillance.

Chapter 3

Datasets related to the thesis

3.1 The SZTAKIBudapest Benchmark for map-based scene analysis

Although several academic benchmarks such as the KITTI [8] and the nuScenes [9] or industrial datasets provided for example by the Honda [23] or the AIMotive [10] company are available regarding 3D urban scene analysis based on onboard AV measurements (Lidar, camera, IMUs, GPS), they do not include dense 3D reference maps. While the nuScenes dataset [9] has a map expansion module, the map thereby refers to only a top-view projection of the scene with semantic layer information.

Our new *SZTAKIBudapest* Benchmark contains RMB point cloud streams captured by a Velodyne HDL 64E 64-beam RMB Lidar sensor with 20 Hz

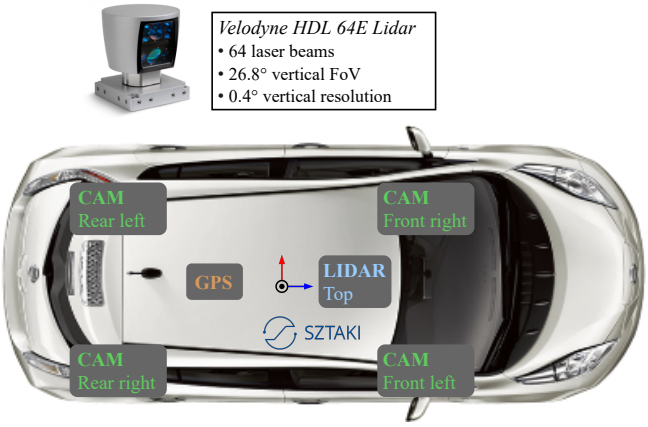


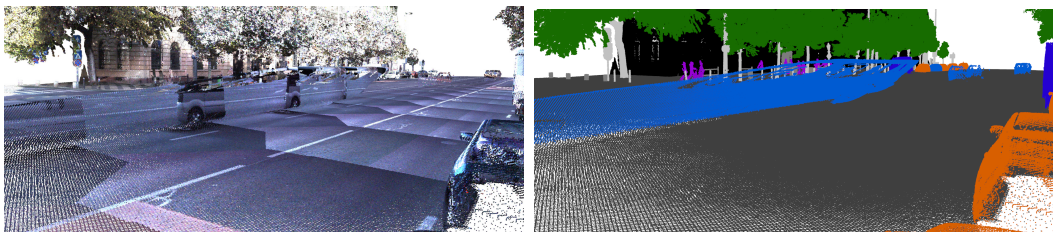
Figure 3.1: The measurement platform.

Figure 3.2: The data collection path for the SZTAKIBudapest dataset.

rotation frequency from different downtown areas of Budapest, where high-density, geo-referred point cloud maps are also recorded by a Riegl VMX450 MLS scanner. The measurement platform of our AV is displayed in Fig. 3.1. PointGrey cameras were also attached to the platform but they were only used for visual verification in this thesis. The Benchmark contains measurements from three different road scenarios, each one covering a path of around 300 meters (see Fig. 3.2) with reference MLS data.

3.1.1 Offline preprocessing of the prior MLS maps

The raw, noisy and dense MLS point clouds may include several measurement segments which do not contain relevant information for vehicle navigation (e.g. many ground points, regions of large building facades, re-located or dynamic objects). First, to reduce the map's size and redundancy, we semantically segmented the raw MLS point clouds as shown in Fig. 3.3 and kept only regions of static object classes (*pillar-like*, *street furniture*, *vegetation* and *facade*), whose appearance do not vary significantly over time, and should be



(a) Captured raw MLS data (b) Segmentation results based on [19]

Figure 3.3: Results of the offline MLS data segmentation, Budapest, Hungary. Color codes: facade (black), ground (dark gray), pedestrian (purple), blurred object (blue), street furniture (hell gray), tall column (mid gray), vegetation (green), vehicle (orange).

also present in *empty* street segments. Note that this step can be performed in an offline pre-processing stage, either in manual or in automatic manner [19]. Next, we extracted object samples from these static class regions by 3D Euclidean clustering [24] and we described each static landmark object with the following parameters:

- Global coordinates (x,y,z) of the object's 3D bounding box corner points (24 parameters)
- Yaw orientation of the object (1 parameter)
- Label of the object class (1 parameter)
- Size of the object's 3D bounding box: width, depth, height and volume (4 parameters)

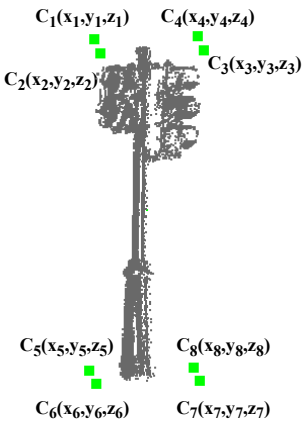


Figure 3.4: Extracted corner points of a tall column object sample from the MLS map.

With this feature extraction, we reduced the number of points describing an MLS object from a few tens of thousands to 30 parameters (see Fig. 3.4), enabling quick high-level access to the map objects for example for fast localization.

3.1.2 Manual annotation and labelling

For enabling quantitative evaluation, we performed manual annotations for each related task described in the following paragraphs to provide pseudo ground truth (GT) information.

For the *point cloud registration* task, we constructed pseudo GT information by manually aligning uniformly selected point cloud frames to the global MLS point cloud at each scenario. While as main drawback here, the manual alignment definitely has some uncertainty, as there is no determinable *,,best"* alignment between the significantly different point sets, we will use this evaluation in combination with other error metrics as well to measure the efficiency of different alignment algorithms, such as the modified Hausdorff distance (MHD) or median point distance (MPD).

For the *global localization* task, pseudo GT information was constructed by manually aligning a full point cloud sequence with consecutive frames to the global MLS point cloud at one scenario that contained several significantly occluded objects.

For the *change detection* task, we annotated uniformly selected Lidar point cloud measurement frames from each test scenario, and provided pseudo GT information in a semi-automatic manner. First, we performed an approximate offline registration between the RMB and MLS frames using the Iterative Closest Point (ICP) [25] algorithm, then we applied an automated nearest neighbor search based classification with a small distance threshold (5 cm) as an initial segmentation result. Thereafter, the labeling of the different change regions (especially on the region borders) in the selected Lidar frames was manually revised using a user-friendly 3D point cloud annotator tool developed by Nagy et al. [19]. Hereby we distinguished three change classes by manual labeling:

- *Dynamic changes* that refer either to moving street objects such as traffic participants or slowly changing objects such as temporal (e.g., barriers) or static scene elements like a re-located bus station or kiosk.

- *Seasonal changes*, which regions are typical for vegetation areas. These regions are segmented as vegetation in the MLS data, and may have modified appearance during the different time periods/seasons.
- *Unchanged regions*, which contain static environment parts. These regions are also present in the MLS data.

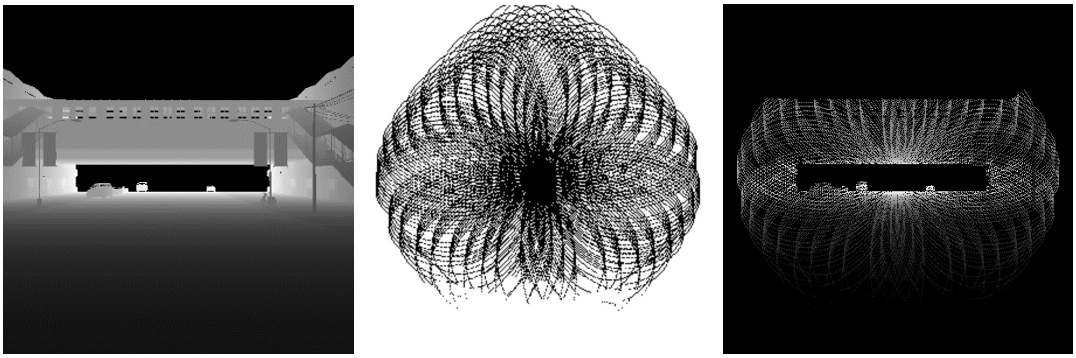
For the *object detection* task, we uniformly selected Lidar point cloud measurement frames from heavy traffic road sections of the test scenarios – in average 5 vehicles and 16 pedestrians in each selected frame –, and labelled the pseudo GT information manually.

3.2 Lidar-densification benchmarks

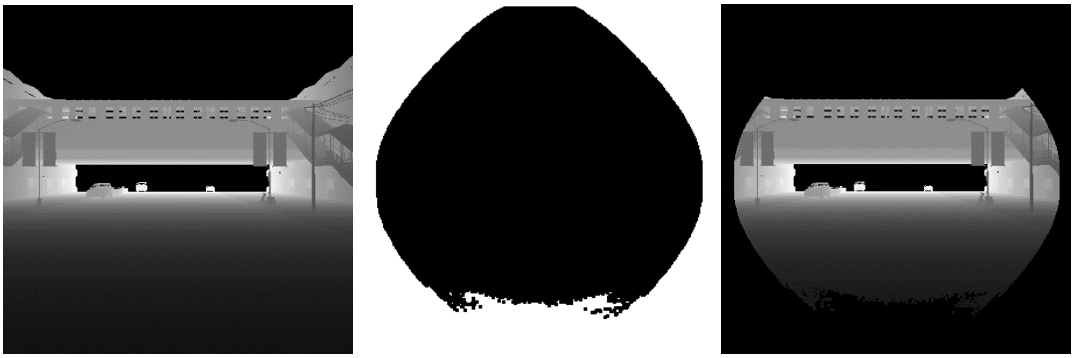
As mentioned in the Introduction, when we are dealing with real NRCS Lidar measurement sequences, it is challenging to provide dense, spatially precise GT depth information for real data due to the independent movements of dynamic objects of the scene including the ego-motion of the robot or vehicle. For this reason, we constructed a synthetic range image dataset called *Livox-CARLA* from a realistic virtual world using the CARLA simulator [26, 27], where we simulated the behaviour of the Livox AVIA NRCS Lidar sensor (Fig. 2.6). The virtual world allows us to extract dense, spatially precise depth information, used as GT for the Lidar's sparse, rosetta patterned samples.

3.2.1 The synthetic LivoxCarla dataset

During data recording, we generated several dynamic objects (vehicles, pedestrians) in the CARLA virtual world. Then, we constructed our capturing platform which was a simulated vehicle that was dynamically moving in the virtual world as realistically as possible. For example, the vehicle was moving with a typical speed and acceleration in an urban environment, it was stopping at red lights or by pedestrian crossing. During data extraction, a synthetic NRCS sensor was placed by default on the front-top of the capturing vehicle and was pointing forwards. To augment the extractable information (e.g., due to varying ground level), the sensor's position was randomly rotated along the up axis by $[-22.5°, 22.5°]$, and its height was randomly adjusted between $[1.5\text{m}, 2.5\text{m}]$. We simulated several runs, from where we randomly exported 5



(a) Example sparse depth image generation by a simulated NRCS Lidar



(b) Example generation of the ground truth depth image

Figure 3.5: The construction process of the LivoxCARLA dataset.

consecutive dense depth images (these were not affected by any distortion or blurring) from the sensor's position, each after 200 ms. These squared images were covering the full field of view of the sensor. Then, to simulate realistic Livox AVIA measurements, the dense depth images were sampled with the rosetta scanning pattern of the Livox AVIA sensor (Fig. 3.5(a)). We extracted these patterns from test measurements with the real sensor device. The ground truth was generated for each sample using the mask with the full field of view of the Livox AVIA sensor (see Fig. 3.5(b)).

Our final *LivoxCARLA* dataset consists of 11726 randomly selected input-output range image pairs, from which 10000 were split for training, 500 as validation and 1226 for testing. Each pair consists of 400×400 images: the input range images were generated with NRCS-characteristics by a Livox AVIA sensor model by 200 ms integration windows (with ca. 40% FoV coverage), while a high-resolution ground truth range image was sampled by each fifth input frame.

3.2.2 The LivoxBudapest real-world dataset

Besides the *LivoxCARLA* dataset, we also collected real measurement sequences from Budapest. In these experiments, we used the Livox AVIA sensor mounted on the front-top of our test vehicle (see Fig. 3.6(a)) on a driving path of total 5.5 kilometers in both speedways and in the city center. Similarly to synthetic data generation, the real test vehicle was continuously moving during the measurements, while many different traffic participants were captured. Although this real dataset, referred as *LivoxBudapest*, does not include GT data, it enables us to validate the effectiveness of the proposed algorithm in real environment.

(a) Sensor setup on the test vehicle (b) Sparse sample data

Figure 3.6: The sensor setup of the capturing platform and a sample sparse frame from the LivoxBudapest test data.

Chapter 4

Map-Lidar fusion for real-time urban scene analysis

This chapter presents a new method for urban scene analysis through fusing Lidar point clouds with significantly different density characteristics. The consecutive steps of the proposed method are briefly summarized in Fig. 4.1. We start with accurately registering the actual RMB measurement to the segmented MLS reference model. Hereby we propose a new object-based coarse-to-fine alignment algorithm, which significantly speeds up the process while

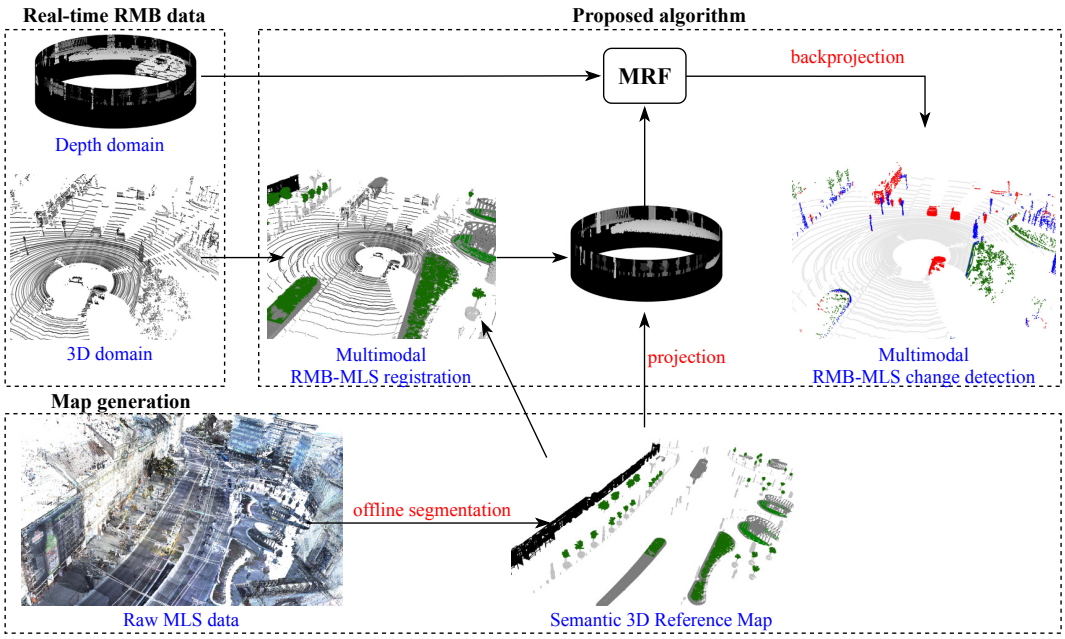


Figure 4.1: The workflow of the proposed method.

keeping the registration accuracy high. Next, we demonstrate an efficient utilization of the introduced registration technique for accurate and robust pose tracking. Then, we perform change detection between the registered RMB and MLS point clouds by a Markov Random Field-based new classification model. In addition, we demonstrate a possible application of the extracted change map for dynamic object detection in Advanced Driving Assistant Systems (ADAS).

4.1 Localization as cross-source point cloud registration

In this section, a novel cross-source point cloud registration algorithm is introduced, which can improve the alignment of the sparse RMB measurements to the dense MLS data, where conventional point level registration or keypoint/segment matching strategies fail. The proposed method assumes that an AV is moving on a (locally) fairly flat surface, and provide a fast 4DoF coarse alignment by matching static objects of the scene, while the exact 6DoF alignment is calculated after a point-level refinement step. The clear advantages of the new method are quantitatively demonstrated against various reference techniques using the *SZTAKIBudapest* Benchmark.

4.1.1 Related work in point cloud registration

In the field of point cloud registration [28], coarse-to-fine alignment strategies are considered as standard approaches [29]. Hereby the coarse-alignment process is usually based on matching corresponding 3D feature points or characteristic primitives extracted from the point clouds [30], while the fine-alignment step aims to minimize distances between points within the overlapping point cloud regions by an iterative process. As discussed in [30], general coarse-alignment methods consist of two main steps:

1. efficient keypoint detection [31], followed by meaningful and discriminative keypoint description [32],

2. executing an iterative process for finding correspondences on a solving platform.

4.1.1.1 Dense-dense registration

For dense MLS data, general handcrafted [31, 33, 34] and learning-based [35, 36] detectors can be directly used to extract keypoints [37]. However, the adaption of such methods often fails for sparse, inhomogeneous and noisy RMB data [38]. The usage of popular feature descriptors based on normal vectors, curvature, density, and pairwise point-to-point distances in local regions [32, 39] cannot enrich the information in cross-modal scenarios, since such parameters are significantly different regarding the RMB and MLS point clouds. Therefore, the lack of pairable feature points in the two compared point clouds results that correspondence-based iterative solving platforms like classical RANSAC [40] approaches or even the very recent TEASER++ [41] cannot provide accurate solutions.

4.1.1.2 Sparse-sparse registration

Further methods [42, 43] focus on region-based point cloud alignment. The method by Douillard et al. [42] solves data mapping by matching segments instead of points across different RMB scans. This approach can efficiently match sparse and small RMB segments with similar point characteristics, but it induces significant computational complexity: around five to fifteen seconds is needed even for such small RMB data pairs, while the running time grows rapidly by using larger point clouds. The SegMap [43] approach matches segments from a few consecutive RMB scans (local map) to a global 3D map with high accuracy. This technique builds first a dynamic voxel grid from the point clouds, then segments the voxels and extracts geometry-based or data-driven features from both segmented clouds. Finally, it finds correspondences in the feature space. However, matching these features correctly is extremely sensitive to the differences in the point cloud characteristics. While according to the experiments in [43], the method works robustly when the global map is generated from the same sensor data as the aligned RMB measurements (e.g. in Simultaneous Localization and Mapping (SLAM) tasks), the method's extension to different sensor modalities is yet to be solved [43]. Using dense MLS point clouds and sparse RMB data, we cannot extract corresponding segments based on similar feature contexts, which fact we experimentally demonstrate in Sec. 4.1.3.

4.1.1.3 Cross-source sparse-dense registration

There are methods addressing directly the problem of registering cross-source point clouds [44], which is more challenging due to varying noise, the large number of outliers and the significantly different density characteristics [45]. To improve on point-level correspondences, the Geometric Constraint Tensor-based registration (GCTR) approach [46] uses triplet point similarities, and solves the optimization problem in the tensor space by an iterative process. However, its computation time remains around 1-2 minutes for matching simple indoor scan pairs, making it unfeasible for real-time applications. Instead of relying on point-point correspondences, the Feature-metric registration (FMR) method [47] solves the cross-source registration problem by minimizing a feature-metric projection error, however, the density differences of its aligned point clouds are less significant than in our RMB-MLS scenario.

4.1.1.4 Registration refinement

For point level registration, which is usually applied in a refinement step after the coarse alignment, the Iterative Closest Point (ICP) [48] is a frequently adopted algorithm with several improvements [49, 50]. However, all of these variants perform local error minimization, thus they demand a high-quality initial estimation for the alignment otherwise they tend to get stuck in local minima. In practice, a position error of several meters does not satisfy this requirement, moreover, the computational time is considerably increased for scans with poor initial alignments.

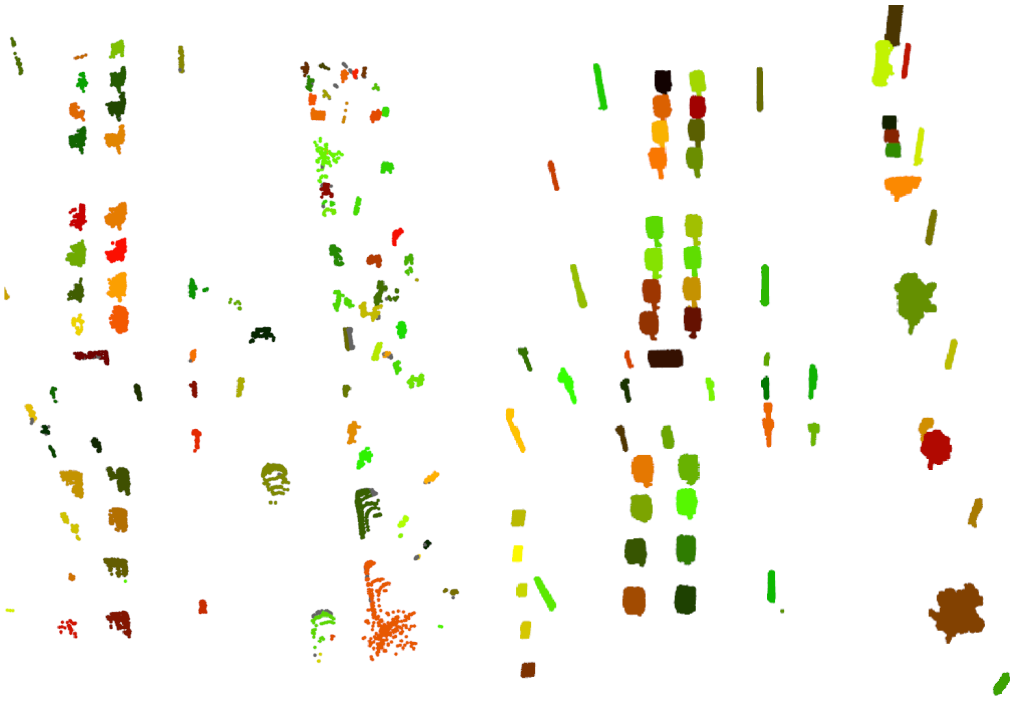
4.1.2 Proposed point cloud registration method

We developed a cross-source RMB to MLS point cloud registration technique to align the RMB point scans to the reference MLS model, which can compensate initial position errors up to several meters, expecting that in dense urban environments with poor GPS coverage, the initial position estimation of an AV might be notably inaccurate. Since we intend to use the RMB point cloud stream recorded by a moving vehicle for real-time decision support, the RMB Lidar point cloud should be fully automatically processed and matched to the MLS model. On the other hand, we can exploit here that in the preliminary segmented MLS point clouds, the ground regions are separated, and

various abstract landmark objects (e.g. pillar-like entities or short street furniture instances) are extracted and labeled (Fig. 4.2(b)) in advance according to Sec. 3.1.1. The steps of the algorithm are discussed in the next parts of this section.

4.1.2.1 Ground removal and object separation in the RMB point cloud

The initial step of the registration process is object segmentation in the RMB Lidar frames. Due to the limited resolution and inhomogeneous density of the RMB point clouds which can mislead even state-of-the-art object detectors [51], we use here a geometry-based approach for point cloud segmentation proposed by Börcs et al. [52]. First, we apply a fast 2D grid-based *ground-obstacle* separation in the RMB input point cloud, by classifying each cell on an estimated ground surface based on local point density and point elevation difference features as follows: We fit a regular 2D grid with 0.2 meter rectangle



(a) Separated objects in a RMB scan (b) Static objects in the MLS data

Figure 4.2: Possible landmark objects in the same scene extracted from the RMB (a) and MLS (b) point clouds for the transformation estimation. Each object candidate is displayed with a different color.

width – parameter optimized to urban environment according to [52] – onto the P_z horizontal plane of the RMB Lidar point cloud's local Euclidean coordinate system. We assign each point to the corresponding cell that contains its projection to P_z. We mark each cell as *ground candidate* where the differences of the observed maximal and minimal point elevation values are lower than 10 centimeters, which condition allows up to 26° ground slope within the cell. Next, we obtain a local elevation map by averaging the point elevation values for each previously marked *ground candidate* cell. To eliminate outlier values from the elevation map, resulted by flat surfaces such as car roofs, we apply a median filter considering the neighbouring ground cells. For the remaining *non-ground* cells – which presumptively contain the obstacles of the scene – the local ground elevation value z_0 is interpolated using the elevation values of the neighbouring ground cells. For each *non-ground* cell, all points with elevation z_p are denoted as non-ground points, where $z_p - z_0 > \tau$ (used $\tau = 10$ centimeters according to [52]).

Next, we cluster all corresponding *non-ground* points of the RMB Lidar frame to separate individual object candidates with a region growing algorithm on the 2D cell map, where empty cells act as stopping criteria. Although in this way, some adjacent objects may be merged together (see in Fig. 4.2(a)) due to the limited resolution of the grid, this approach is around a hundred times faster [52] than conventional 3D Euclidean clustering algorithms such as the Connected Component Analysis [24], while it can efficiently separate even nearby objects. In our approach, for ensuring fast processing and robustness, this step does not perform any attempt on object classification: as a result, we only extract a set of 3D blobs that may represent various moving or static obstacles in the urban environment.

4.1.2.2 Transformation estimation

Next, we estimate an optimal transformation to align the sparse RMB scan to the MLS reference model. As a key idea, instead of aligning the raw point clouds, we aim to register the frames via an object-level voting algorithm, which matches the previously extracted *landmark objects* of the MLS data and the *separated objects* from the RMB scan (Fig. 4.3). Here a major challenge is that in the automatically segmented RMB Lidar frames one should expect plenty of spurious objects which cannot be matched to the MLS landmarks:

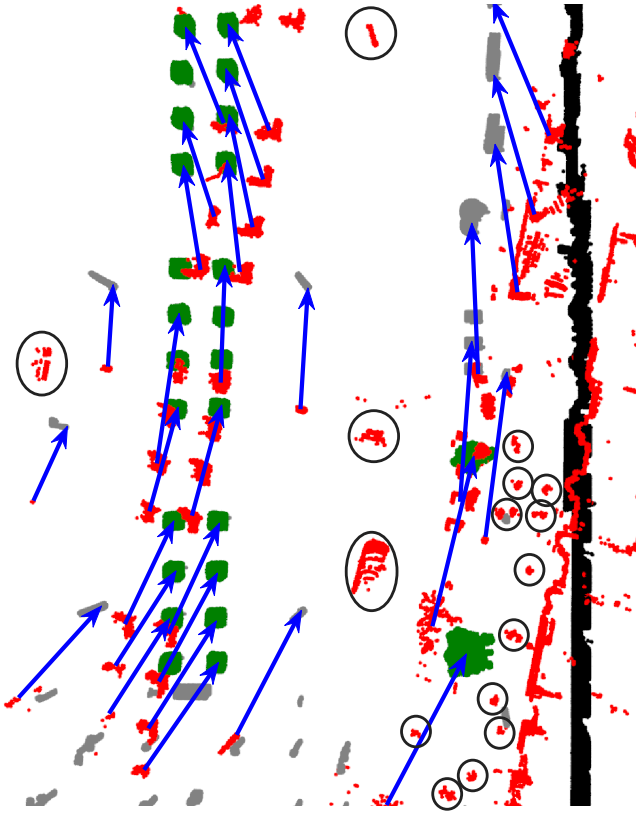


Figure 4.3: Displaying the corresponding RMB-MLS object pairs estimated by the proposed method, based on the generalized Hough transform-based schema [53]. RMB points are displayed with red, object pairs are marked by ***blue arrows****, some outlier objects are circled by black. (For clear visualization, a scene sample with relatively few moving objects has been chosen here.)*

the RMB frames may include many dynamic objects, and further artifacts can be caused by partially extracted entities due to occlusions. For quickly matching the two object sets which typically contain many objects (often a few dozen) with possibly a large ratio (up to 80%) of outliers, we turned to a robust generalized Hough transform-based technique, that proved earlier efficient for different sorts of complex assignment problems such as fingerprint minutiae matching [53].

First, we rely on the available GPS signals to initially align the actual RMB point cloud frame to the reference coordinate system of the MLS data. This initial alignment is usually notably inaccurate, which we enhance first at a coarse level, by searching for an optimized rigid transformation with a 3D translation and a rotation component between the point clouds. The translation component $(\Delta x, \Delta y, \Delta z)$ compensates for the GPS-based offset error,

while – based on experiments – the rotation component can be fairly modeled by a single rotation value (θ) around the upright axis of the vehicle. As a consequence, we model the optimal coarse transform by a 4×4 homogeneous matrix as shown in the following equation:

$$\mathbf{T}_{\Delta x,\Delta y,\Delta z,\theta} = \begin{bmatrix} \cos\theta & \sin\theta & 0 & \Delta x \\ -\sin\theta & \cos\theta & 0 & \Delta y \\ 0 & 0 & 1 & \Delta z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Next, we find the optimal parameters of this $\mathbf{T}_{\Delta x,\Delta y,\Delta z,\theta}$ transformation in three steps, presented in the following subsections.

a) Semantic compatibility constraints As described in Chapter 3.1, the MLS-based 3D reference map contains separated tall *pillar-like* and shorter *street furniture* object point clouds, which can be used as landmark objects by the AVs. For these landmarks, we prescribe the following compatibility constraints:

- a detected RMB object (see Sec. 4.1.2.1) is compatible with a *pillar-like* MLS object if the vertical side length of its bounding box is more than twice of its width and depth parameters,
- a RMB object is compatible with a *street furniture* object if the ratio of their bounding volumes is between $[0.75, 1.25]$.

In the upcoming transformation estimation step, we will only consider the votes of compatible object pairs, to decrease the effects of outliers and speed up the process. Note that the impacts of false matches generated by dynamic objects of the RMB point clouds will be eliminated later through global parameter optimization.

b) Keypoint selection Since the detected objects are *not pointwise*, appropriate keypoint extraction is a critical step which process should remain robust in the considered crossmodal RMB-MLS scenario, where several objects are only partially scanned, the object appearances in the RMB frames are different from the corresponding MLS objects, and they may also vary scan by scan. We have formerly investigated various keypoint selection strategies and experienced that an 8-keypoint strategy [54] provided the best result,

where, as keypoints, we choose the eight corners of the objects' 3D bounding boxes.

c) Optimal parameter estimation We approximate the transformation which aligns the RMB and MLS point clouds via compatible keypoint pairs. First, we collect the extracted *RMB* and *MLS* objects into two object sets marked by $\mathcal{O}_{\text{RMB}}$ and $\mathcal{O}_{\text{MLS}}$, where each object is described by 8 keypoints (i.e., bounding box corners). Then we adopted the generalized 4D Hough transform [53] to obtain an optimal transformation between the *RMB* and *MLS*-based keypoint sets, by a voting algorithm (Fig. 4.3).

To restrict the search space of transformations, we allow maximum offsets of $\pm 60°$ for rotation (θ), ± 12 meters for planar translation (Δx and Δy), and ± 2 meters for vertical translation (Δz). As required by the Hough schema [53], the parameter space is discretized with step sizes of 0.2 meters for translation and 0.25° for rotation, which choice enables both reasonable coarse alignment and quick computation. The votes of the possible parameter quartets are accumulated in a 4D array $\Phi[\Delta x, \Delta y, \Delta z, \theta]$, which is initialized with zero values.

The optimization step searches for possible correspondences between the

Algorithm 1 The proposed coarse registration method.

procedure CoarseAlignment($\mathcal{O}_{\text{RMB}}, \mathcal{O}_{\text{MLS}}$)
  Reset the 4D accumulator array Φ
  for all $o_{\text{RMB}}, o_{\text{MLS}} \in \mathcal{O}_{\text{RMB}} \times \mathcal{O}_{\text{MLS}}$ **do**
    if compatible($o_{\text{RMB}}, o_{\text{MLS}}$) **then**
      for $i = 1 : 8$ **do**
        $k^i_{\text{RMB}} \leftarrow o_{\text{RMB}}$
        $k^i_{\text{MLS}} \leftarrow o_{\text{MLS}}$
        for all $\theta \in [-60°, 60°]$ **do**
          $k^{i*}_{\text{RMB}} \leftarrow Rot\theta \cdot k^i_{\text{RMB}}$
          $[\Delta x, \Delta y, \Delta z] \leftarrow k^i_{\text{MLS}} - k^{i*}_{\text{RMB}}$
          $\Phi[\Delta x, \Delta y, \Delta z, \theta] \leftarrow \Phi[\Delta x, \Delta y, \Delta z, \theta] + 1$
  $\Delta x^*, \Delta y^*, \Delta z^*, \theta^* \leftarrow FindMaximum(\Phi)$
  $T \leftarrow \Delta x^*, \Delta y^*, \Delta z^*, \theta^*$
  return T

keypoints of all *compatible* object pairs $(o_{\mathrm{RMB}}, o_{\mathrm{MLS}}) \in \mathcal{O}_{\mathrm{RMB}} \times \mathcal{O}_{\mathrm{MLS}}$. For a given keypoint couple k_{RMB}, k_{MLS} we increase the evidence of all $\mathbf{T}_{\Delta x,\Delta y,\Delta z,\theta}$ mappings, which move k_{RMB} to k_{MLS}. More specifically, for every possible $\theta' \in [-60°, +60°]$ value, we rotate k_{RMB} by θ' first, then the following $[\Delta x', \Delta y', \Delta z']^T$ offset is computed:

$$\begin{bmatrix} \Delta x' \\ \Delta y' \\ \Delta z' \end{bmatrix} = k_{\mathrm{MLS}} - \begin{bmatrix} \cos\theta' & \sin\theta' & 0 \\ -\sin\theta' & \cos\theta' & 0 \\ 0 & 0 & 1 \end{bmatrix} k_{\mathrm{RMB}}$$

Next, we vote for the calculated $T_{\Delta x',\Delta y',\Delta z',\theta'}$ transform so that we increase the $\Phi[\Delta x', \Delta y', \Delta z', \theta']$ element of the accumulator array by one (see Algorithm 1). After iterating through the whole parameter space, the optimal $T_{\Delta x^*,\Delta y^*,\Delta z^*,\theta^*}$ transform is defined as follows:

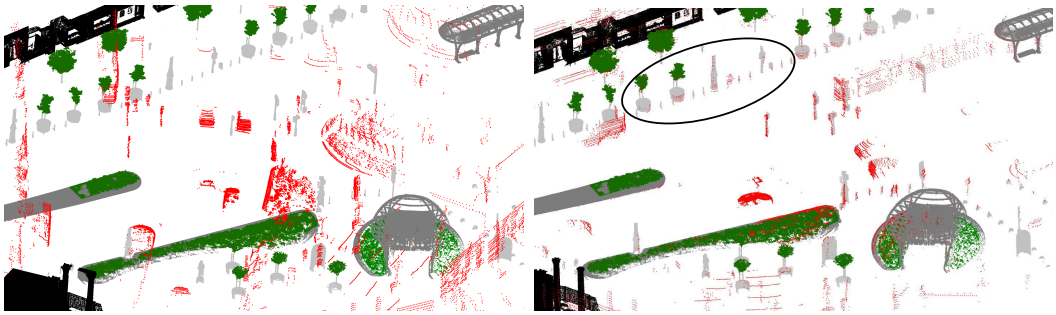
$$(\Delta x^*, \Delta y^*, \Delta z^*, \theta^*) = \underset{\Delta x,\Delta y,\Delta z,\theta}{\operatorname{argmax}} \Phi[\Delta x, \Delta y, \Delta z, \theta]$$

4.1.2.3 Registration refinement

While acknowledging its robustness, the precision of the object-based registration technique is affected by the discretization step of the translation and rotation parameters, constraints on the modeled rigid transformation, and by issues of object bounding box fitting on partially detected objects (Fig. 4.4(c)). However, this coarse alignment step can provide an efficient initialization for a point level refinement algorithm, such as the ICP [48], which we execute for point cloud segments corresponding to aligned object pairs only. The final transformation is taken as:

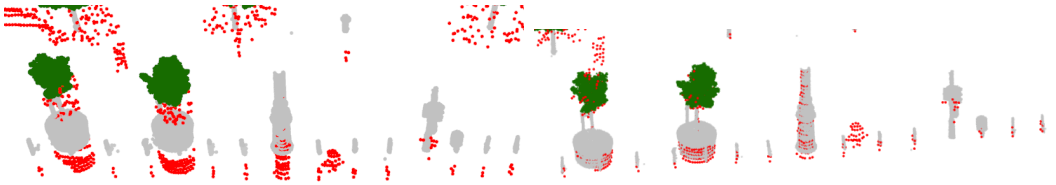
$$\mathbf{T}_{final} = \mathbf{T}_{\mathrm{ICP}} \cdot \mathbf{T}_{\Delta x^*,\Delta y^*,\Delta z^*,\theta^*}$$

Although the application of the ICP algorithm induces additional computation, by reducing the number of alignable points through object compatibility criteria checking, and by ensuring low initial alignment error by the proposed coarse registration algorithm, the whole process needs remarkably – with 1-2 orders of magnitude – less computational time compared to matching raw complete point cloud scans, while keeping the registration accuracy high (Fig. 4.4(b), (d)).



(a) Initial alignment, $t \approx 4.17$ m translation and $\theta \approx -51°$ rotation error

(b) Results of the proposed registration algorithm



(c) Results of the coarse alignment in case of an uneven road surface

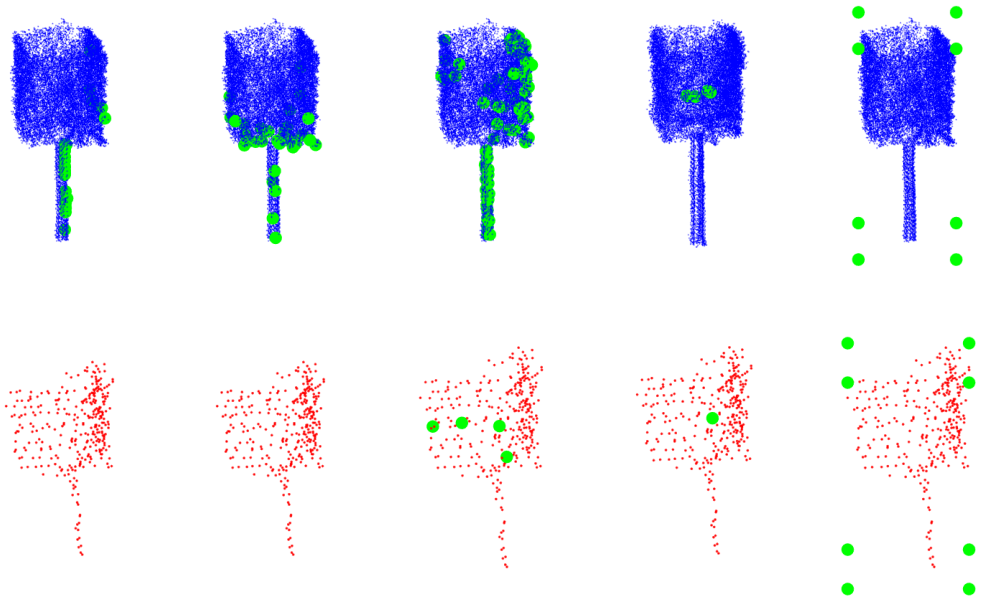
(d) Results after the point level refinement

Figure 4.4: Results of the proposed point cloud registration algorithm. Subfigures (c) and (d) refer to the same area circled by black in subfigure (b). Color codes: RMB points are shown with red, the segmented MLS regions are marked by various colors depending on their semantic classes: facade (black), vegetation (green), street furniture (dark grey), pillar-like column (light grey).

4.1.3 Evaluation

This section presents various numerical and qualitative results on the introduced *SZTAKIBudapest* Benchmark, which demonstrate the efficiency of the proposed cross-source point cloud registration technique and its superiority versus the state-of-the-art reference approaches.

First, to justify the need for developing a new registration algorithm for the crossmodal RMB and MLS data alignment task, we demonstrate the limitations of methods based on existing keypoint selection strategies (discussed in Sec. 4.1.1) adopted for the sparse RMB Lidar and dense MLS point clouds, respectively. As shown in Table 4.1, the numbers of keypoints extracted from given object clusters (such as short and tall pillars, tree trunks, etc.) are in different orders of magnitude in point clouds captured by the different sensors, while Fig. 4.5 (a)-(d) show visually that we are unable to detect the same or even similar keypoints from the sparse RMB and dense MLS scans of a 3D



(a) Harris 3D (b) SIFT 3D (c) ISS 3D (d) RSKDD-Net (e) Proposed

Figure 4.5: 3D keypoint selection strategies for registration on the same object stored in the MLS (top) and RMB point cloud (bottom).

street object. The RSKDD-Net [36] strategy extracts visually the less different keypoints from the two point clouds, however, their similarity is still not enough for proper registration (see also Table 4.3 and 4.4).

Among the region-based methods, we also tested the SegMap [43] approach adopted to the RMB and MLS data (Fig. 4.6). Although the method extracts correctly the point cloud segments of most standalone static landmark objects in the RMB data (red points), it fails to find correct correspondences between segments of the RMB and MLS point clouds (black arrows) due to their local

Method	Extracted keypoints per cluster	
	MLS cloud	RMB cloud
Harris 3D	10-20	-
SIFT 3D	30-50	-
ISS 3D	100+	1-10
RSKDD-Net	5-20	0-5

Table 4.1: Typical numbers of the extracted keypoints by different handcrafted and learning-based methods in MLS and RMB point clouds

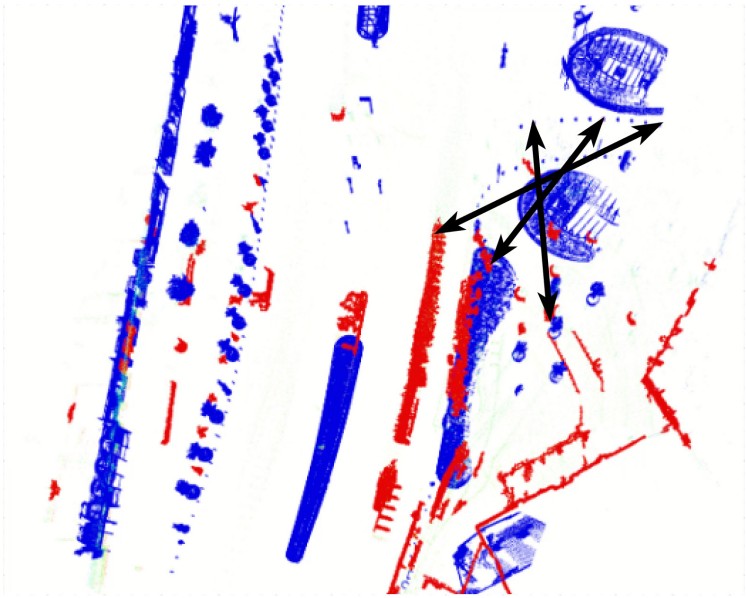


*Figure 4.6: False correspondences (**black arrows**) using the SegMap approach [43] between the sparse RMB local map (displayed by red) and dense MLS global map (shown by blue).*

contextual differences (especially in point density).

We evaluated the proposed Hough space-based registration technique on the *SZTAKIBudapest* dataset. The efficiency of the algorithm can be demonstrated by qualitative examples, even in cases of very large initial alignment errors (see Fig. 4.4). For quantitative analysis, the *SZTAKIBudapest* Benchmark contains manually aligned pseudo GT transformation matrices, which enables direct comparison in the rigid transformation's independent parameters (three rotation and three translation components), similarly as in the work of Park et al. [55]. However, this manner of comparison has some limitations, as in several cases the error of only a single parameter can significantly distort the final transformation. Moreover, the manual registration process definitely includes some uncertainties (up to 10-20 cm translation, 0.5-1° rotation). For this reason, we calculated two further point-to-point distance metrics [56] to measure how well our transformed RMB frames can fit the global MLS segments. First, we considered the *Modified Hausdorff Distance (MHD)* [57, 58] between the $\mathcal{P}_{\text{RMB}}$ and $\mathcal{P}_{\text{MLS}}$ obstacle clouds:

$$Q_{\text{MHD}}(\mathcal{P}_{\text{RMB}}, \mathcal{P}_{\text{MLS}}) = \frac{1}{\#\mathcal{P}_{\text{RMB}}} \sum_{p \in \mathcal{P}_{\text{RMB}}} \min_{q \in \mathcal{P}_{\text{MLS}}} ||p - q||$$

Here $\#\mathcal{P}$ marks the number of points in set $\mathcal{P}$. However, for some scenes with

Scene	Initial avg. error	Q_{MHD} [m]			Q_{MPD} [m]			Comp. time*
		init.	coarse	fine	init.	coarse	fine	
Deák	1.4m, -54.98°	5.558	0.681	0.631	3.140	0.059	0.019	1.583
Fővám	2.2m, 53.04°	4.853	1.092	0.956	2.147	0.161	0.019	2.317
Kálvin	3.6m, 38.53°	5.504	0.872	0.745	3.384	0.285	0.038	3.272
Average	2.4m, 48.85°	5.298	0.882	0.779	2.873	0.164	0.025	2.594

*with sequential CPU implementation, in [s]

Table 4.2: Quantitative results of the proposed registration algorithm by MHD and MDP rates

many dynamic objects (including large vehicles), the Q_{MHD} metrics proved to be less relevant regarding the evaluation of the results, due to many outlier values which were accumulated. Therefore, we also turned to an alternative measure referred to as *Median Point Distance (MPD)*, which ranks the points in $\mathcal{P}_{\text{RMB}}$ by their $\min_q ||p-q||$ values and the median distance over all $p \in \mathcal{P}_{\text{RMB}}$ is calculated:

$$Q_{\text{MPD}}(P_{\text{RMB}}, P_{\text{MLS}}) = \underset{p \in P_{\text{RMB}}}{\text{Med}} \min_{q \in P_{\text{MLS}}} ||p - q||$$

Note that the calculation processes of both MHD and MPD are unsupervised, thus they can be considered as relative quality measures rather than metric error values.

Table 4.2 displays the MHD and MPD rates for the three different scenes of the *SZTAKIBudapest* Benchmark, calculated at different stages of the proposed two-step registration algorithm. We can see that both quality values are significantly reduced by the process, and in all test scenarios, the final observed MPD error values are around 1-3 cm. Note that high registration accuracy can also be verified by qualitative analysis (see Fig. 4.4).

Table 4.3 and 4.4 presents a detailed quantitative comparison between the proposed method and different reference approaches for point cloud registration on the *SZTAKIBudapest* Benchmark. Quality measures provided here comprise both the differences in the transformation's independent parameters (Table 4.3) and the point distance-based MHD and MPD error rates (Table 4.4). For each method, the GPS position was taken as an initial alignment. Based on the experiments, the one-step least square optimization [28] can decrease the overall point distances but completely fails to find the optimal po-

Method	Absolute error compared to pseudo GT transformation					
	Δr_x [°]	Δr_y [°]	Δr_z [°]	Δt_x [m]	Δt_y [m]	Δt_z [m]
GPS baseline	1.831	1.492	48.784	1.485	5.999	1.103
SegMap	-[a]	-[a]	-[a]	-[a]	-[a]	-[a]
RSKDD-Net	94.037	6.670	76.047	9.466	15.065	1.241
Least-Square	89.335	9.117	85.655	5.516	5.493	0.627
ICP	1.951	2.170	30.778	2.023	5.411	0.608
RANSAC with FPFH	19.921	2.434	7.263	3.518	7.422	0.853
TEASER++	5.522	1.664	12.350	1.613	5.477	1.162
FMR	9.062	4.544	19.105	5.137	5.234	1.180
Proposed coarse	1.831	1.492	**0.812**	**0.199**	**0.273**	**0.333**
Proposed with ICP	**0.598**	**0.959**	**0.333**	**0.069**	**0.154**	**0.183**

[a] failed to find correspondences

Table 4.3: Comparative evaluation of various point cloud registration methods and the proposed approach compared to GT

sition and orientation when the density characteristics of the two point clouds significantly differ. Similarly, the RSKDD-Net [36] method cannot improve the original alignment due to the lack of extractable keypoint correspondences (see Fig. 4.5(d)). The high rotation errors of these methods [28, 36] along the x (Δr_x) and z-axes (Δr_z) in Table 4.3 correspond to the large percent of test cases where the alignable point clouds were falsely flipped horizontally or vertically, respectively. The FMR [47] method found false feature matches derived from the local point-distribution differences of the point sets, resulting in higher translation errors along the x and y-axes. The ICP algorithm [48] – which aims to minimize locally the errors – failed to converge from the poor GPS-based position in around 75% of the test frames, and for the remaining cases, the accurate ($<$ 10 cm) registration needed 3-4 minutes computation time for a single frame.

For the further global registration techniques, like the RANSAC [40] and TEASER++ [41] have been applied as follows: first, the two point clouds were uniformly downsampled, then the Fast Point Feature Histogram (FPFH) [32] feature descriptors were calculated to extract the local geometric properties of the point neighbourhoods and the correspondences were detected by querying the nearest neighbours in the 33-dimensional FPFH feature space. Although these approaches could reduce both the position and orientation errors, they

Method	Distance-based evaluation		Computation time [s]
	Q_{MHD} [m]	Q_{MPD} [m]	
GPS baseline	5.298	2.873	-
SegMap	-[a]	-[a]	-[a]
RSKDD-Net	2.968	2.789	**0.048**[b]
Least-Square	2.510	2.176	0.364
ICP	1.483	0.700	239.768[c]
RANSAC with FPFH	2.318	1.877	0.793
TEASER++	1.818	1.134	3.849
FMR	3.392	3.387	25.596[c]
Proposed coarse	**0.882**	**0.164**	**0.102**
Proposed with ICP	**0.779**	**0.025**	2.594[c]

[a] failed to find correspondences [b] GPU-accelerated impl. (on CPU: 1.0844 [s])
[c] Sequential CPU impl. (GPU-acceleratable)

Table 4.4: Comparative evaluation of various point cloud registration methods and the proposed approach by distance-based errors and computation time.

turned out to be less accurate than the proposed method due to many outliers, which were eliminated by the proposed approach with the voting scheme. Note that the RANSAC-based registration also rotated the point clouds upside down in around 10% of the test frames, resulting in relatively higher rotation error along the x-axis (Δr_x).

Regarding the computation time, the coarse alignment step of the proposed algorithm can work with around 10 fps on a desktop environment with an Intel Core i7-7700K 4.2 GHz CPU, while the fine alignment steps need currently 2-4 seconds, both on sequential CPU implementation (see the last column of Table 4.4). However, a recent study [59] show that using parallel implementation of the correspondence search in ICP, one can reduce its computational time with one order of magnitude on multi-core CPU (OpenMP) and with two orders of magnitude on GPU (CUDA). These hardware-accelerated implementations can make the proposed algorithm eligible for real-time applications.

4.2 Lidar pose tracking by matching static objects

In this section, we propose a real-time, robust pose estimation and tracking technique in temporally occluded urban environment for AVs using sparse RMB Lidar and low accuracy GPS measurements, with respect to prior high density localization maps obtained from MLS point clouds. This approach relies on the registration algorithm which was introduced earlier in Sec. 4.1. First, assuming that we can extract the local ground information from the map, we estimate the 3DoF pose of the vehicle from the RMB-MLS registration results. Then, we effectively integrate the estimated pose information into a dynamic vehicle model based Kalman filter. The advantage of the new method is qualitatively and quantitatively demonstrated in a large-scale urban scenario [3], using the *SZTAKIBudapest* Benchmark.

4.2.1 Related work in pose tracking

Robot or vehicle localization and tracking given a prior map is a hot topic in the literature [60]. In general, we can distinguish methods addressing global localization or pose estimation (when no prior pose is available), and pose tracking, when the vehicle starts from a known pose which is updated over time.

In the field of pose tracking, the majority of existing methods uses Lidar odometry information [61] by incrementally aligning consecutive Lidar point cloud measurements mostly using a variant of the Iterative Closest Point (ICP) [25] algorithm, and determining the relative pose of the moving vehicle at each iteration. Some methods integrate Lidar odometry with IMU sensors for more accurate results [62, 63]. As these methods integrate small incremental motions over time, they are bound to drift-effect in large-scale scenarios, which is typically reduced by loop closure detection.

Tackling the problem of global localization in large-scale urban environment with poor GPS coverage, the pose estimation problem can be described as a point cloud registration between the Lidar-measurements and map data, starting from a poor initial alignment [1]. As a general overview of point cloud registration algorithms were already discussed in Sec. 4.1.1.

The closest solution to our addressed scenario is the SegMap [43] technique,

which detects wall segments of the corresponding Lidar and map regions, and describes these regions using geometric or data-driven features. On the other hand, similar features are hard to extract for segments captured with different sensor modalities, as experimentally shown in Sec. 4.1.3, while the lack of close wall segments (i.e., in wide open spaces, or due to occlusions) can mislead this method as well. Following a different approach, we aim to match pillar-like objects (poles, traffic lights, signs, etc.) of the scenes in real time, which sorts of objects are typically present in urban regions. In addition, we also introduce an efficient pose tracking method to tackle featureless measurement frames (e.g., due to temporal occlusions by moving objects).

4.2.2 Proposed pose tracking method

We developed a real-time, robust pose estimation and tracking technique for AVs with respect to prior MLS localization maps, using sparse onboard RMB Lidar and low-accuracy GPS measurements. As a preliminary step, we efficiently extract and describe the static objects of the MLS data by their geometric and semantic properties. This process was described earlier in Sec. 3.1.1. Next, for estimating the optimal pose of the vehicle, we adopt a simplified version of the robust transformation algorithm proposed in Sec. 4.1 to align the RMB Lidar data and the extracted static objects of the MLS map. Finally, from the optimal transformation – assuming locally planar surfaces – we extract the 3DoF pose of the vehicle (x,y,θ), which parameters we track by a constant velocity model-based position-only-measured (POM) Kalman filter to effectively deal with temporal occlusions.

4.2.2.1 Pose estimation of the moving vehicle

First, similarly as in Section 4.1, we use the available, usually notably inaccurate GPS signal for initially positioning the actual RMB point cloud frame's center in the global coordinate system of the MLS map. Assuming that the local ground plane information is available from the map, we search for an optimized rigid transformation with a 2D translation and a rotation component between the point clouds. The translation component $(\Delta x, \Delta y)$ compensates for the originally unknown position error of the GPS sensor, while – based on experiments – the rotation component can be approximated by the

yaw rotation angle ($\Delta\theta$). In summary, we model the optimal transform as follows:

$$\mathbf{T}_{\Delta x,\Delta y,\Delta\theta}\begin{pmatrix}x\\y\end{pmatrix}=\begin{bmatrix}\cos\Delta\theta & -\sin\Delta\theta\\ \sin\Delta\theta & \cos\Delta\theta\end{bmatrix}\begin{pmatrix}x\\y\end{pmatrix}+\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix}$$

Next, we find the optimal parameters of this $\mathbf{T}_{\Delta x,\Delta y,\Delta\theta}$ transformation via extracted keypoint pairs. First, we collect the extracted pillar-like *RMB* Lidar and *MLS* map objects into two sets denoted by $\mathcal{O}_{\text{RMB}}$ and $\mathcal{O}_{\text{MLS}}$. Then, we describe each object candidate by 8 keypoints (i.e. bounding box corners), and adopt the generalized 3D Hough transform to determine the optimal transformation between the *RMB* and *MLS*-based keypoint sets, by a voting algorithm.

First, for limiting the parameter space, we allow maximum offsets of $\pm 60°$ for the yaw rotation ($\Delta\theta$) and ± 12 meters for planar translation (Δx and Δy) to tackle with the GPS inaccuracies. As required by the Hough schema, we discretize the transformation space between the minimal and maximal allowed values of each parameter, using 0.4 meters for the translation components and 0.5° degrees for rotation. This setup enables both reasonably accurate resolution and quick computation. Next, we allocate a three-dimensional array $A[\Delta x, \Delta y, \theta]$ with zero initial values to summarize the votes of the possible parameter triplets.

During the voting process, we search for possible keypoint correspondences between all *pillar-like* object pairs $(o_{\text{RMB}}, o_{\text{MLS}}) \in \mathcal{O}_{\text{RMB}} \times \mathcal{O}_{\text{MLS}}$. For a given keypoint couple k_{RMB}, k_{MLS} we add a vote for all possible $\mathbf{T}_{\Delta x,\Delta y,\Delta\theta}$ transforms, which map k_{RMB} to k_{MLS}. More specifically, we iterate over all the discrete $\Delta\theta$ values, and for each $\Delta\theta'$ we rotate k_{RMB} by $\Delta\theta'$ first, and calculate the corresponding translation vector $[\Delta x', \Delta y']^T$ as follows:

$$\begin{bmatrix}\Delta x'\\ \Delta y'\end{bmatrix}=k_{\text{MLS}}-\begin{bmatrix}\cos\Delta\theta' & -\sin\Delta\theta'\\ \sin\Delta\theta' & \cos\Delta\theta'\end{bmatrix}k_{\text{RMB}}$$

Next, we vote for the calculated $\mathbf{T}_{\Delta x',\Delta y',\theta'}$ transform so that we increase the $A[\Delta x', \Delta y', \theta']$ element of the accumulator array by one. After iterating through the whole parameter space, the optimal $T_{\Delta x^*,\Delta y^*,\theta^*}$ transform can be extracted as follows:

$$(\Delta x^*, \Delta y^*, \Delta\theta^*) = \underset{\Delta x,\Delta y,\Delta\theta}{\operatorname{argmax}} A[\Delta x, \Delta y, \Delta\theta]$$

Finally, we make an acceptance decision of calculated transform based on a minimum number of votes:

$$\mathcal{A}(\mathbf{T}) = \text{true if and only if } A[\Delta x^*, \Delta y^*, \Delta\theta^*] > t$$

We experimentally set $t = 5$, which means that either one static object is matched by both upper and lower corner points or at least two static objects are paired. In case of an accepted transform, the estimated pose of the vehicle can be calculated as follows:

$$\begin{aligned} x^* &= x_{\text{GPS}} + \Delta x^* \\ y^* &= y_{\text{GPS}} + \Delta y^* \\ \theta^* &= \theta_{\text{GPS}} + \Delta\theta^* \end{aligned}$$

4.2.2.2 Pose tracking

Although we experienced that the above pose estimation method works robustly even in sparse scenes covering only a few (5-10, depending on the scene characteristics) pillar-like objects, its accuracy is limited in scenarios without a sufficient number of matchable landmark object pairs. To overcome this problem, we track the estimated pose parameters by a constant velocity (CV) model based Kalman filter, whose true state vector is defined as follows:

$$\mathbf{x_t} = \begin{pmatrix} x_t & y_t & \theta_t & v_{xt} & v_{yt} & w_{\theta t} \end{pmatrix}^T$$

Here x_t, y_t and θ_t are the planar position and yaw orientation and v_{xt}, v_{yt} and $w_{\theta t}$ are the velocities of the vehicle, respectively. As the CV model assumes permanent velocity within a short observation period, the model's dynamics can be considered as follows:

$$\mathbf{x_{tk}} = \mathbf{\Phi x_{tk-1}} + \mathbf{w_k},$$

Here $\mathbf{x_{tk}}$ denotes the true state at time kT, T is the sampling interval determined by the applied RMB Lidar sensor's spin rate, $\mathbf{w_k}$ is the process noise,

and $\mathbf{\Phi}$ is the transition matrix from kT to $(k+1)T$, which is defined as:

$$\mathbf{\Phi} = \begin{pmatrix} 1 & 0 & 0 & T & 0 & 0 \\ 0 & 1 & 0 & 0 & T & 0 \\ 0 & 0 & 1 & 0 & 0 & T \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Next, we integrate the Lidar-perceived planar position and yaw orientation values into this model dynamics, so that after each *accepted* transformation ($\mathcal{A}(\mathbf{T}) = \text{true}$), we use the estimated poses (x^*,y^*,θ^*) as new measurements as follows:

$$\mathbf{z_k} = \mathbf{H}\mathbf{x_{tk}} + \mathbf{v_k},$$

Here $\mathbf{z_k} = (x^*, y^*, \theta^*)$ denotes the measurement vector, $\mathbf{H}$ denotes the measurement matrix, and $\mathbf{v_k}$ is the measurement noise. As our model is position-only-measured, $\mathbf{H}$ is:

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Finally, we sequentially predict and estimate the state vectors based on the previous state values and measurements via the Kalman filter equations:

$$\tilde{\mathbf{x}}_\mathbf{k} = \mathbf{\Phi}\hat{\mathbf{x}}_{\mathbf{k-1}}$$
$$\hat{\mathbf{x}}_\mathbf{k} = \tilde{\mathbf{x}}_\mathbf{k} + \mathbf{K_k}(\mathbf{z_k} - \mathbf{H}\tilde{\mathbf{x}}_\mathbf{k})$$

where $\tilde{\mathbf{x}}_\mathbf{k}$ and $\hat{\mathbf{x}}_\mathbf{k}$ are the predicted, respectively estimated state vectors by the Kalman filter, while $\mathbf{K_k}$ denotes the Kalman gain that minimizes the errors in the estimated positions and velocities. If the Lidar-based estimation of the pose transformation is *not accepted* (i.e. $\mathcal{A}(\mathbf{T}) = \text{false}$ in Sec. 4.2.2.1), we only execute the prediction step, while the state vector re-estimation is skipped.

4.2.3 Evaluation

We evaluated the proposed pose tracking technique on a heavily occluded sequence from the *SZTAKIBudapest* Benchmark, in a pathway of around 0.6 km. During quantitative evaluation, we compared the results of the proposed model to available pseudo GT information, generated through manually aligning the RMB Lidar frames to the global MLS point clouds. As evaluation

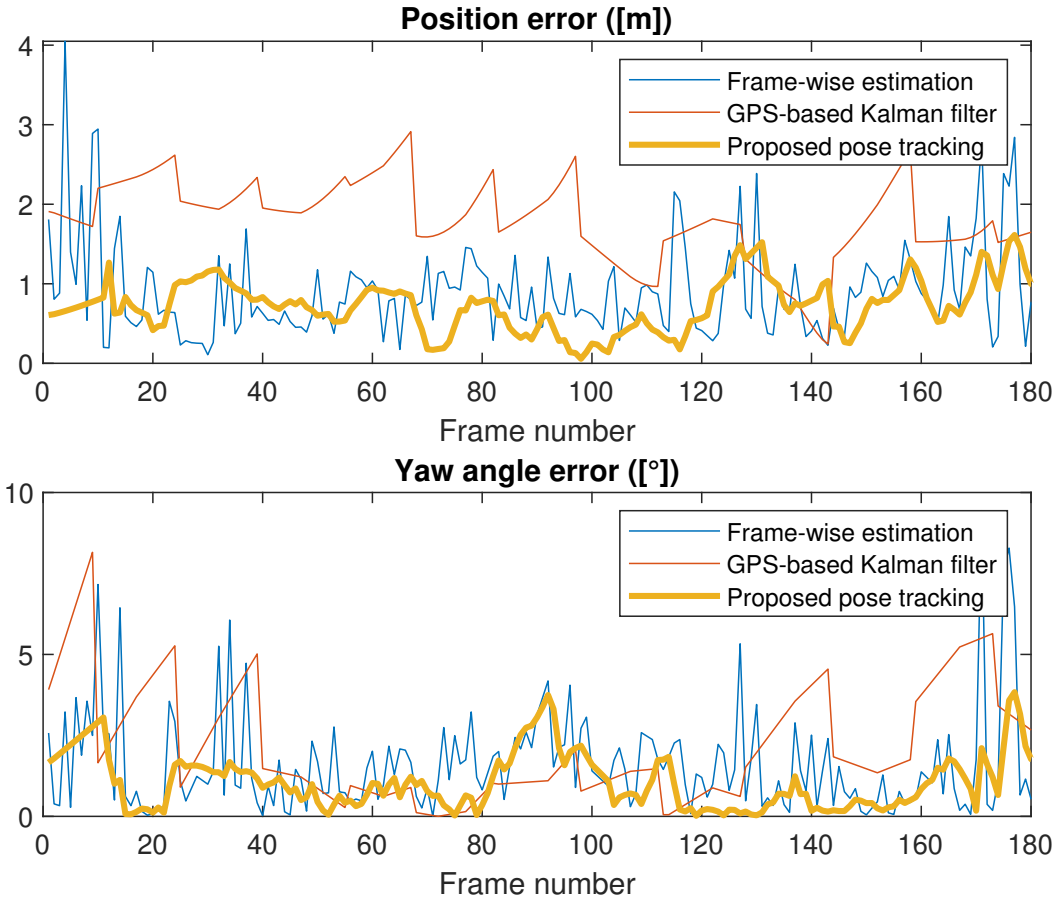


Figure 4.7: Position (D) and orientation (d_θ) error of the tracked poses versus ground truth information, demonstrating the superiority of the proposed method.

metrics, we calculated the mean absolute error (MAE) for each estimated pose parameter: position errors along the x-axis (d_x) and y-axis (d_y), and the yaw orientation error (d_θ). For two-dimensional position error, we also calculated the average Euclidean distance (D) between the estimated and GT planar positions (x,y) of the vehicle. For comparative experiments, we developed a baseline, only GPS-based Kalman filter and adopted as Lidar-based frame-wise pose estimation method the previously proposed registration algorithm (Sec. 4.1) without tracking, besides the proposed improvement. The overall numerical results are summarized in Table 4.5.

Fig. 4.7 displays a sequence of error rates calculated for 180 consecutive time frames from the test scenario, covering a driven path of approximately

Figure 4.8: Estimated trajectories of the different methods and the ground truth path.

300 meters. During this drive, frames containing large moving objects (tram, bus) with significant occlusions were recorded, while the trajectory of the ego vehicle was turning to left of around 20 degrees. In the pose estimation step, 20% of the calculated transforms were dropped due to the lack of enough pairable objects. The planar trajectories estimated by the different methods can be also visually compared in Fig. 4.8.

Method	d_x [m]	d_y [m]	D [m]	d_θ [°]
Raw GPS	3.4214	3.2428	4.9288	29.768
GPS-only Kalman filter	1.5184	0.7691	1.8004	2.1962
Frame-wise pose estimation	0.5852	0.5628	0.9029	1.6131
Proposed method	**0.4850**	**0.4349**	**0.7304**	**1.0592**

Table 4.5: Quantitative summary of the pose parameter errors

From the results of Table 4.5 and Fig. 4.7, we can conclude that using the frame-wise Lidar-based pose estimation we can significantly improve the accuracy of the GPS-only positioning, reducing the average location error from around 5 meters to 1 meter. However, the value of the error is still strongly fluctuating frame-by-frame without considering the vehicle dynamics (see Fig. 4.7). The proposed joint method can largely overcome this artifact by efficiently integrating the Lidar-perceived pose information into the Kalman filter based dynamic model of the moving vehicle, achieving an average global position error of around 70 centimeters and orientation error around 1° in approximately only 40-50 milliseconds.

Regarding the computation time of the whole workflow, the proposed method can operate in real-time, as it runs with 20-25 fps on an Intel Core i7-7700K 4.2 GHz CPU without parallelization. The presented result may also serve as a fast and accurate initial alignment for an ICP-based [25] point level registration algorithm in occluded environment (which can also run in real-time with parallel implementation), that may decrease the location error to a few centimeters.

4.3 RangeMRF: Range image-based cross-source change detection

In this section, an efficient Markov Random Field-based [64] change extraction step is proposed between the registered RMB and MLS point clouds, which exploits the fact that due to geometric considerations of mapping with the given sensor configuration, the essence of the problem can be solved quickly in the 2D range image domain without information loss. The clear advantages of the new method are quantitatively demonstrated against various reference techniques using the *SZTAKIBudapest* Benchmark.

4.3.1 Related work in point cloud change detection

In the field of point cloud-based change detection, the majority of the existing methods can be adopted for MLS-MLS data comparison tasks [65, 66], where the two point clouds are captured with the same laser scanner [65] showing similar and locally homogeneous density characteristics. In practice, two

different scannings of the same scene never detect the same surface points of the objects, thus there are point-level deflections between the scanned models even if the surfaces are perfectly aligned. However, using the same MLS sensor, change detection can be approximated by pointwise comparison steps in the 3D space, performing a preliminary uniform voxel-based downsampling [67], or applying locally adaptive [68] or parametric [69] radius thresholds. Nevertheless, the usability of any point-level distances strongly depends on the density of the point clouds [66], and it can be largely misleading during the analysis of multimodal data, like in an RMB-MLS scenario. Alternative methods like [70] use segment-level comparison, where an object is marked as change if a given percentage of its segment points have no neighbours. Voelsen et al. [71] combine this approach with segment classification to integrate semantic information in the change detection process. These methods demand accurate object extraction (and detailed classification) from the point sets, which is challenging in sparse RMB data [52]. To handle irregular point density, Xiao et al. [66] combine point-to-triangle distance calculation, ray-tracing, and occupancy grid generation. Although this method can effectively deal with occlusions and penetrable MLS objects, the triangulation step may mean a bottleneck in terms of computational speed and robustness, especially in noisy point cloud segments. Instead of using rays, Liu et al. [72] perform an occupancy technique along a regular voxel grid, and identify the changed regions based on the inconsistent voxels. As the main drawback, the above approaches work similarly for all point segments: On one hand, with parameter settings yielding high sensitivity, they produce many false-positive detections for sparse and noisy segments of the RMB scans. On the other hand, with low sensitivity, they might ignore crucial regions containing only a few points (e.g., pedestrians with only 10-40 points), which issue will be demonstrated in Sec. 4.3.3.

4.3.2 Proposed change detection method

After accurately registering the RMB-MLS measurements, the proposed method, called RangeMRF, detects changes in 2D range images derived from the point clouds. As key advantages of using a compact range image representation, the proposed method is notably quick, meanwhile, it can robustly handle the significantly different characteristics of the two point sets so that we define the RMB data-based (I_{RMB}) and the MLS range images (I_{MLS}) over

the same discrete pixel lattice R. Since we search for changes caused by moving or removed/added scene objects, before range image mapping, we remove ground points, yielding two obstacle clouds denoted by $\mathcal{P}_{\text{RMB}}$ and $\mathcal{P}_{\text{MLS}}$, which we wish to compare.

Exploiting the principle of operation of RMB Lidar sensors, the obstacle cloud $\mathcal{P}_{\text{RMB}}$ of a considered RMB frame can be represented as a range image I_{RMB} in a straightforward way. Here the laser emitters and sensors are vertically arranged, and each sensor scans the environment along a circular trajectory of 360° (see Fig. 1.2). Within a time frame, the consecutive range measurements of the ith sensor are stored in the ith row of the I_{RMB} image. This transform is geometrically equivalent to converting the representation of the point cloud from the 3D Descartes to a spherical polar coordinate system, where the polar direction and azimuth angles correspond to the horizontal and

(a) Raw range image generated from an RMB point cloud scan

(b) Interpolated range image (I_{RMB}) from subfigure (a)

(c) Reference range image projected from the MLS data (I_{MLS})

(d) Vegetation label propagated from the semantic MLS data (L_{MLS})

Figure 4.9: Example input data of the proposed RangeMRF model. Subfigures (a)-(c) are depth images where brighter pixels denote closer distance, and black pixels contain no measurements. Subfigure (d) displays semantic labels of the MLS data, where vegetation is marked by green.

vertical pixel coordinates, and the distance is encoded in the corresponding pixel value. As a result of discretization, the range values of some image pixels might be undefined, which issue is handled by interpolation (Fig. 4.9(a), (b)). Apart from these issues, with using an appropriate, sensor-specific image resolution (used 1042×64 in the thesis), the conversion of the point clouds to 2D range images is reversible, without causing information loss.

We derive the I_{MLS} background range map from the 3D MLS measurement ($\mathcal{P}_{\text{MLS}}$) by projection. Relying on the previously introduced point cloud registration step, we emit simulated rays into the MLS point cloud from the estimated actual center position of the AV mounted RMB Lidar, and project the point distances to a spherical surface, on which we stretch an image lattice R having the same size and resolution parameters as the I_{RMB} range image. Hereby, utilizing that the reference MLS cloud is semantically segmented, we assign to each pixel $s \in R$ besides its calculated range value $I_{\text{MLS}}(s)$ a binary vegetation indicator label $L_{\text{MLS}}(s) \in \{V, \not{V}\}$ based on the semantic class of the corresponding projected MLS point (Fig. 4.9(c), (d)).

In the next step, we construct a Markov Random Field (MRF)-based model for efficient estimation of the changes between the RMB Lidar's I_{RMB} and the MLS-based I_{MLS} range images, with also considering the L_{MLS} vegetation indicator map (Algorithm 2). The output change map of the MRF model is denoted by C_{RMB}.

By assuming the presence of purely solid-shaped objects in the scene (such as vehicles, pedestrians, traffic signs), change detection could be considered as a binary classification problem with *foreground* (i.e. changes) and *background* (unchanged regions) classes, with applying background subtraction for foreground extraction. It is essential here to recognize even small changes between

Algorithm 2 The main steps of the proposed RangeMRF change detection algorithm, using a Markov Random Field model in the range image domain.

procedure RangeMRF($\mathcal{P}_{\text{RMB}}, \mathcal{P}_{\text{MLS}}$)
  $I_{\text{RMB}} \leftarrow \mathcal{P}_{\text{RMB}}$
  $I_{\text{MLS}}, L_{\text{MLS}} \xleftarrow{projection} \mathcal{P}_{\text{MLS}}$
  $d, \delta \leftarrow (I_{\text{RMB}}, I_{\text{MLS}}, L_{\text{MLS}})$
  $C_{\text{RMB}} = MRF(d, \delta)$
  return C_{RMB}

the inputs, such as pedestrians close to a bus station, therefore, this detection process must work with high sensitivity. However, we experienced that in this approach several false/irrelevant change predictions may occur, especially in vegetation regions, whose appearance widely varies in time, across different seasons. To overcome this artifact, we introduced a third class, called *seasonal change*. In summary, in the proposed change detection step, we distinguish three classes: (a) seasonal changes (S) in vegetation regions, (b) foreground changes (F) caused by moving objects or changed/re-located static street furniture elements, and (c) unchanged background (B). By handling the vegetation areas in a specific manner with reduced sensitivity, we may lose some information in extreme situations such as pedestrians hiding in trees or bushes, however, these cases are very rare and less relevant for analyzing traffic scenarios. Furthermore, the proposed method will be able to sharply recognize pedestrians standing near stations or facades, while also eliminating several false hits in vegetation areas.

Formally, the goal is to perform the following mapping:

$$C_{\mathrm{RMB}} = \mathrm{MRF}\left(I_{\mathrm{RMB}}, I_{\mathrm{MLS}}, L_{\mathrm{MLS}}\right),$$

where $C_{\mathrm{RMB}}(s) \in \{F, S, B\}$ for all $s \in R$.

To set up the MRF energy function, we define two distance values first which can give us reliable information about separating the different classes:

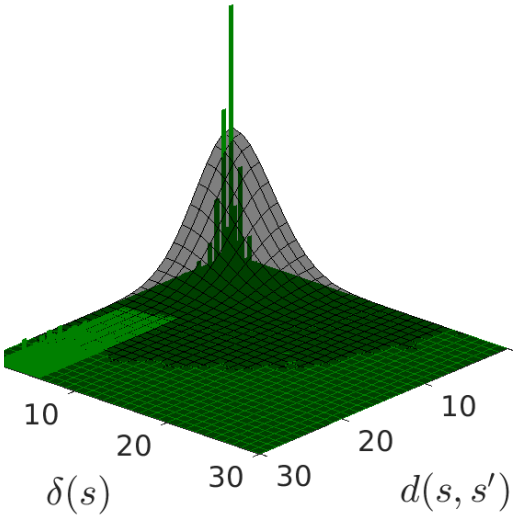
- The *geometric distance* $d(s, s')$ represents the depth difference between the corresponding range values of a given pixel s in the I_{RMB} and s' in the I_{MLS} images, respectively, which is calculated as follows:

$$d(s, s') = |I_{\mathrm{RMB}}(s) - I_{\mathrm{MLS}}(s')|.$$

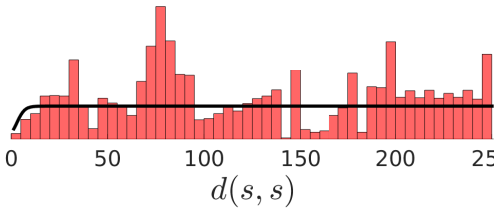
- The *vegetation distance* $\delta(s)$ informs us if the current pixel s is likely in vegetation regions. This parameter is calculated as the L2 distance between the locations of a given s pixel in I_{RMB} and the nearest pixel s' in I_{MLS} which has a vegetation label:

$$\delta(s) = \sqrt{(s_x - s'_x)^2 + (s_y - s'_y)^2},$$

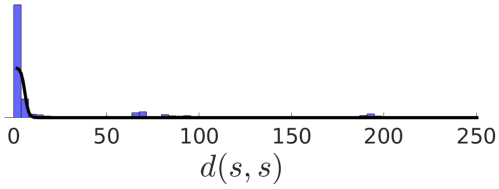
where pixel $s'(s'_x, s'_y)$ fulfills that $L_{\mathrm{MLS}}(s') = V$, and the L2 distance calculated as $\sqrt{(s_x - s'_x)^2 + (s_y - s'_y)^2}$ between (s, s') is minimal among all nearby pixels.



(a) Seasonal change (S)



(b) Foreground change (F)



(c) Unchanged background (B)

*Figure 4.10: The distribution of distance values (d,δ) and the applied fitness functions (marked with **black**) for each change class.*

Next, for every pixel in the range image lattice R, we define fitness scores for each class as functions of the above-described d geometric and δ vegetation distance values, in order to measure how a given pixel s fits the different classes using expert knowledge-based soft constraints. The types of the functions were chosen by experiments, by investigating the empirical distributions of the distance values within different regions of the *SZTAKIBudapest* dataset, which is demonstrated in Fig. 4.10.

Dynamic foreground regions (F) should typically have increasing fitness scores with growing geometric distances ($d(s,s)$) between the corresponding pixel range values in the I_{RMB} and I_{MLS} maps. This soft constraint can be modeled by a *logistic* function with zero midpoint ($d_0 = 0$) and two parameters: maximum value (L) and steepness (k):

$$\mathcal{F}_F(s) = \frac{L}{1 + e^{-k(d(s,s)-d_0)}}.$$

On the contrary, static background (B) pixels have high fitness for small geometric distances, which allows us to use the same *logistic* function with a negative steepness parameter:

Parameter/Class	F	B	S
Geometric distance	$d(s,s) \gg 0$	$d(s,s) \to 0$	$d(s,s') \approx 0$
Vegetation distance	-	-	$\delta(s,s') \approx 0$
Fitness function	logistic	logistic	2D Gaussian
Parameters	L, k	$L, -k$	σ_d, σ_δ
Parameter dimension	2	0^*	2

*The same L, k parameters are used for the **F** and **B** classes.

Table 4.6: Fitness functions and their parameters

$$\mathcal{F}_B(s) = \frac{L}{1 + e^{k(d(s,s)-d_0)}}$$

Seasonal change regions (S) have high fitness scores if and only if both the geometric distance term and the spatial distance to the closest vegetation pixel are near to zero. Formally, this constraint can be described by a *2D Gaussian* function with zero means ($\mu_d, \mu_\delta = 0$) and predefined small standard deviation (σ_d, σ_δ) parameters (Fig. 4.10):

$$\mathcal{F}_S(s) = \frac{1}{2\pi\sigma_d\sigma_\delta} e^{-\left[\left(\frac{d(s,s')-\mu_d}{2\sigma_d}\right)^2 + \left(\frac{\delta(s)-\mu_\delta}{2\sigma_\delta}\right)^2\right]}$$

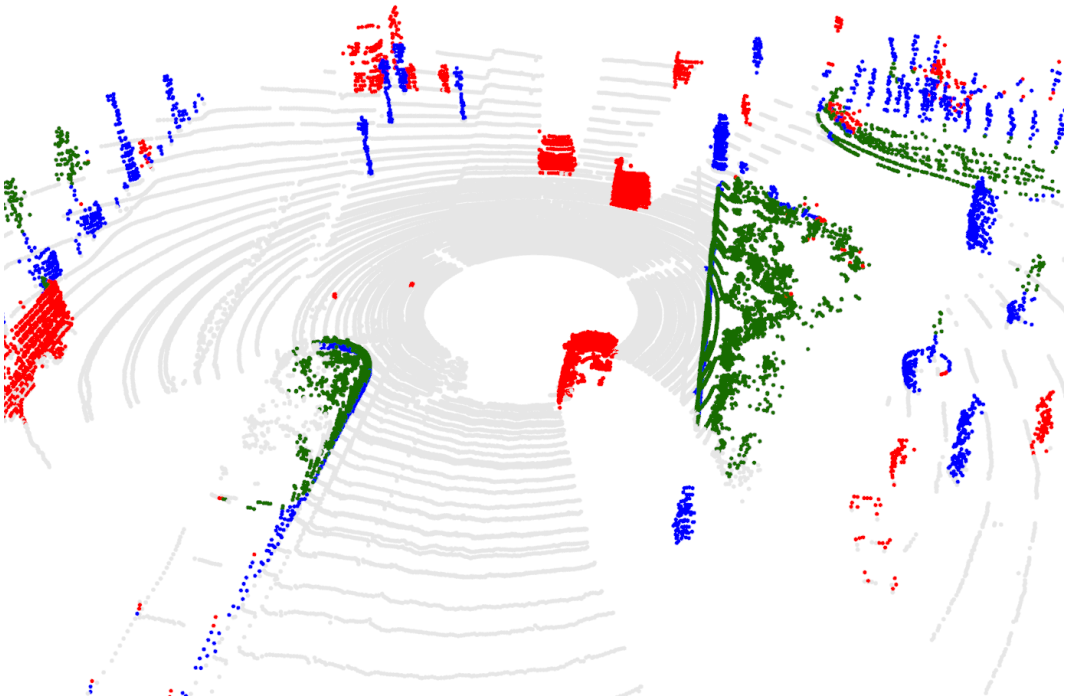
where pixel s' denotes the nearest vegetation pixel to a given pixel s. The chosen fitness functions and parameters are also summarized in Table 4.6.

The proposed change detection algorithm assigns a unique label $l_s \in \{F, S, B\}$ to every $s \in R$ pixel of the lattice, which minimize a Potts-like energy function:

$$E = \sum_{s \in R} -\log(\mathcal{F}_{l_s}(s)) + \sum_{s \in R} \sum_{s^* \in N_s} \beta \cdot 1\{l_s \neq l_{s^*}\}$$

where $\beta > 0$ is responsible for obtaining smooth, connected regions in the segmented image, and N_s denotes the eight-neighbourhood of pixel s. The MRF's output 2D change map is taken as $C_{\text{RMB}}(s) = l_s \; \forall s \in R$.

For the minimization of the MRF energy function E, a quick graph-cut based optimization method has been adopted [73], which provides a high-quality three-class change map ($\{F, S, B\}$) in real time as demonstrated in Fig. 4.11(b). As the last step, the labels available in the range image domain should be projected back to the corresponding points of the RMB Lidar point cloud (Fig. 4.11(a)). Fig. 4.11(b) and (a) show results for the same scene in the 2D and 3D domains, respectively.



(a) Change map in the 3D space

(b) Change map: output of the MRF in the range domain (C_{RMB})

Figure 4.11: Result of the change detection process (a) in the 3D space and (b) in the range image domain, about the same area. The pixels/points for static background are displayed by blue, for dynamic change by red, and for seasonal change by green.

4.3.3 Evaluation

This subsection presents various numerical and qualitative results on the introduced *SZTAKIBudapest* Benchmark, which demonstrate the efficiency of the proposed cross-source change detection technique and its superiority versus the state-of-the-art reference approaches. During evaluation, each change detection algorithm takes as input registered RMB and MLS point clouds, as a result of the previously discussed point set alignment. Since the *SZTAKIBudapest* Benchmark contains pseudo GT labels for the changed regions, we have also performed here quantitative evaluation at point level, so that we compared the labeled output RMB data to the manual annotation of the point cloud.

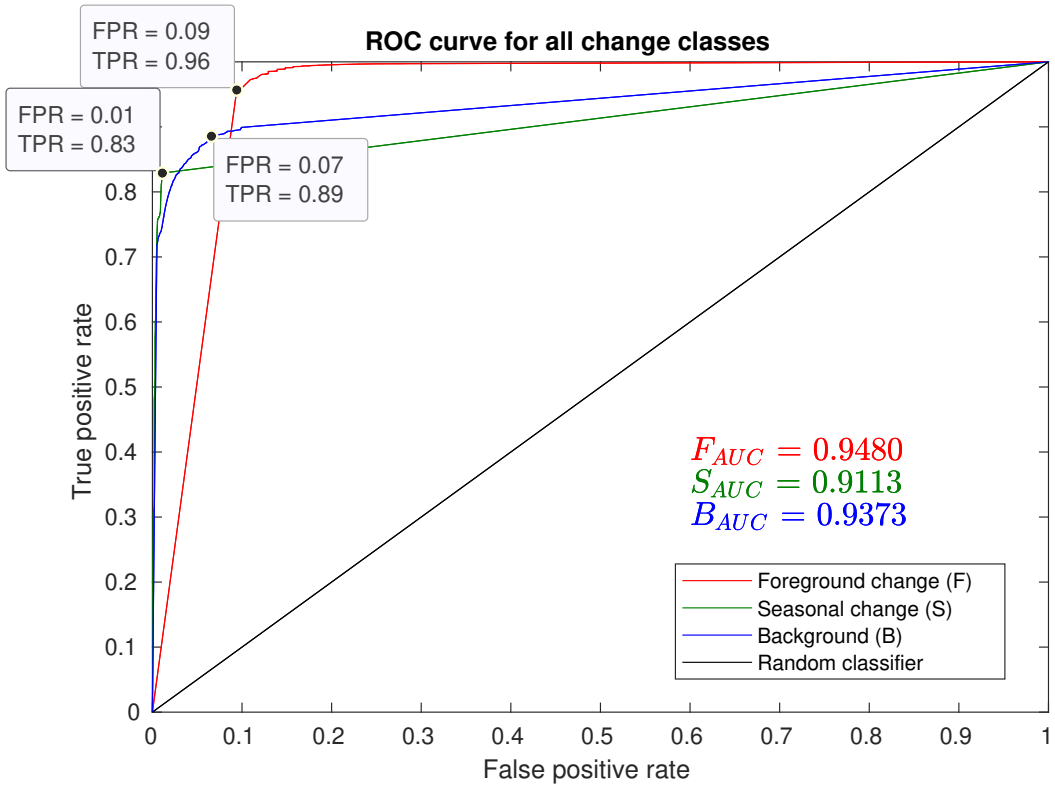


Figure 4.12: The behavior of the proposed method using multiple parameter combinations (L,k,σ_d, σ_δ,β).

4.3.3.1 MRF parameter settings

During the evaluation, first, we compared more than 30000 parameter combinations in a multi-level five-dimensional grid search in order to optimize the parameters (L,k,σ_d, σ_δ,β) of the MRF model. The ROC curve of the results of different setups is summarized in Fig. 4.12. We achieved the best performance using the MRF smoothness parameter $\beta = 0.5$, the logistic function with $L = 0.01$, $k = 2.0$ and the Gaussian function with deviation parameters $\sigma_d = 1.4$ and $\sigma_\delta = 2.5$.

4.3.3.2 Implementation of the reference methods

Since to our best knowledge the proposed RangeMRF method is the first approach dedicated to the RMB-MLS crossmodal change detection task, we selected and adopted reference techniques from the methods described in Sec. 4.3.1, which were proposed earlier for comparing registered MLS point clouds.

First, we implemented a fixed radius ($r = 15$ cm based on [70]) nearest neighbour (NN) search [70] between segments of the two point clouds. We

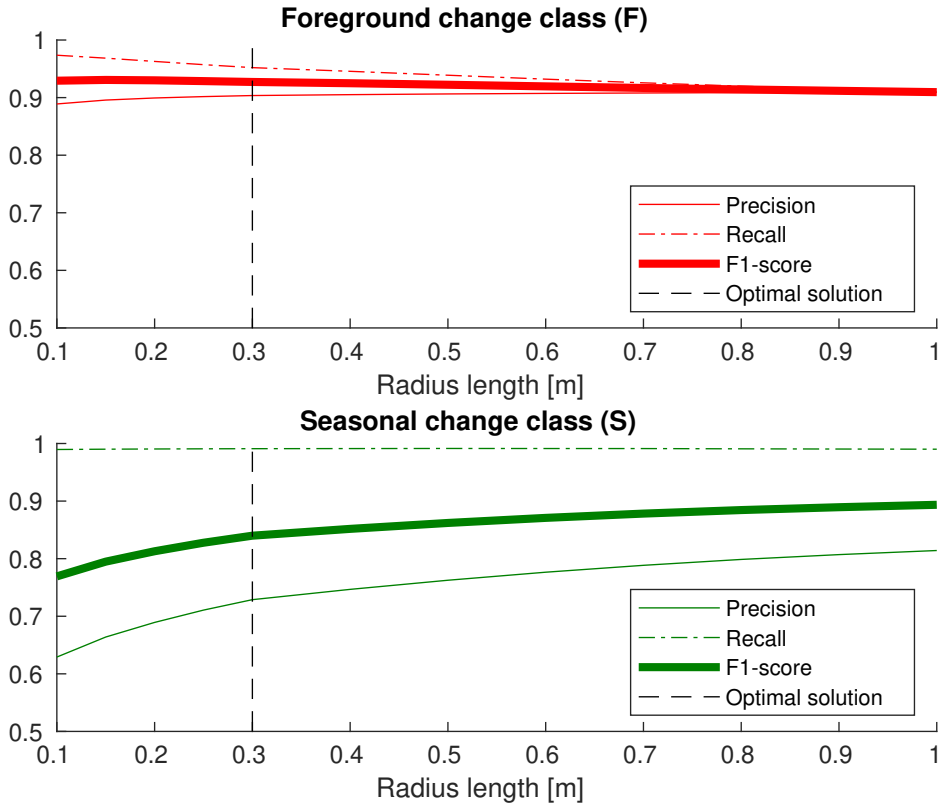


Figure 4.13: The behavior of the point level radius nearest neighbour (NN) search [68, 69] using multiple r parameters. As a balanced solution, $r = 30$ cm was chosen.

labeled each segment of the RMB data as *foreground (F)* if 25% of its points have no neighbours, otherwise, we propagated the label of the neighbouring MLS points to the corresponding segment (*background (B)* or *vegetation (V)*). Next, we implemented the radius search at point level which is used by [68,69]: Each point of the RMB is labeled as F if the distance to the nearest MLS point is higher than r, otherwise the label of the neighbouring point is propagated. We tested this method with multiple r radius lengths (see Fig. 4.13). At small radius, the method failed to compensate the measurement noise and the variance of vegetation areas in the RMB data, while with higher radius values it produces a significantly increasing number of false-negative points. For numerical comparison, we applied an optimally balanced solution for both classes ($r = 30$ cm). To overcome the above-mentioned trade-off, we also implemented a point-to-triangle search based on [66]: for each point in the RMB data, we constructed a triangle surface from its 10 nearest points and

calculated the distance ($d = 30$ cm applied based on [66]) of the given point to the nearest triangle.

Finally, we also fitted a 3D voxel array [72] to the aligned point clouds and classified the given RMB points as *foreground* (F) if its voxel was empty in the MLS point cloud. Otherwise, the label of the MLS point was propagated. We evaluated this technique with various w voxel size parameters (Fig. 4.14) and experienced that for the RMB-MLS data samples it provides the most efficient results with $w = 100$ cm settings, which also gave the best trade-off between the accuracy and computational time.

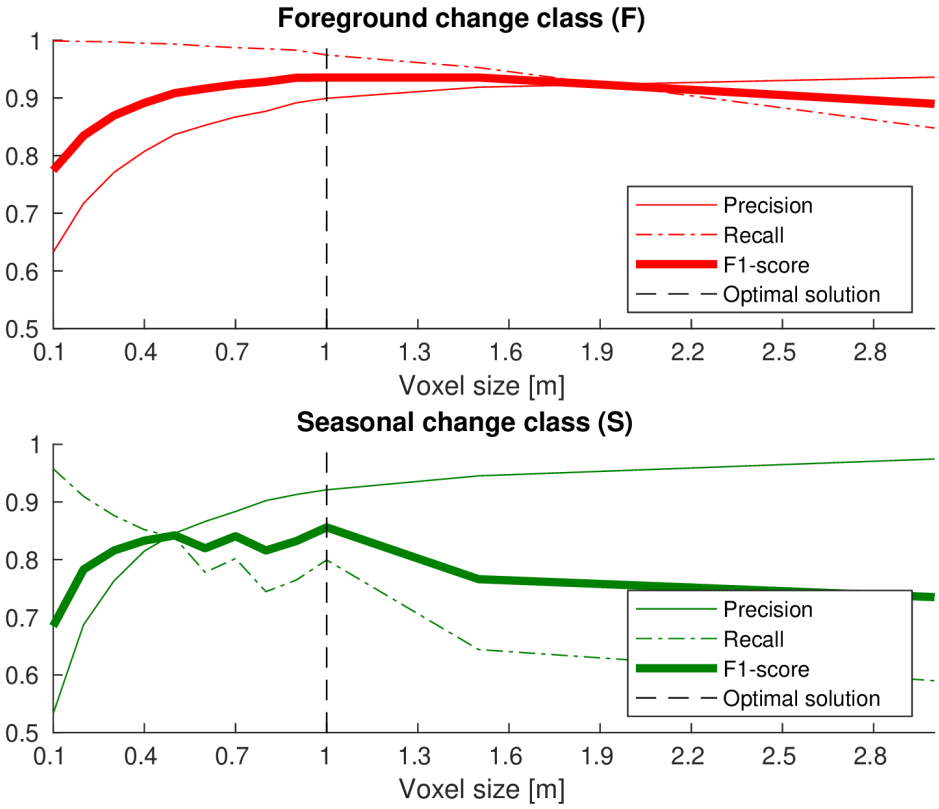


Figure 4.14: The behavior of the voxel-based reference method [72] using multiple w parameters. The optimal solution for both classes is with $w = 100$ cm.

4.3.3.3 Comparative results

While we have observed that in point cloud regions with several large, non-contacting vehicle objects, the reference techniques have similarly high

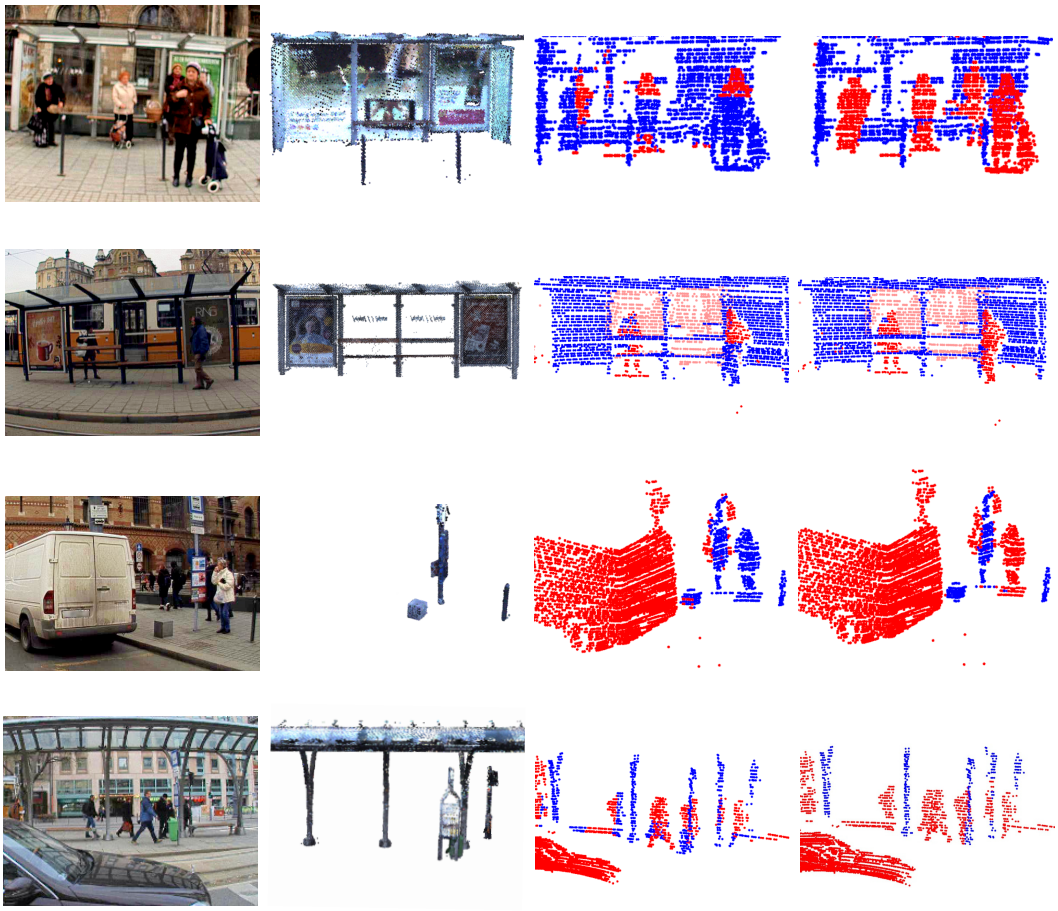
accuracy as the proposed method, we have experienced notable advantages of the RangeMRF model on challenging situations captured in cluttered sidewalk regions, containing multiple pedestrians and a wide variation of static objects, such as benches, boxes, columns (see Fig. 4.15). For this reason, we have focused the quantitative comparison of the methods on the cluttered street regions. Results are summarized in Table 4.7 displaying precision, recall, and F1-score metrics calculated for all approaches.

In these experiments, we have measured the lowest F1-scores with the segment level NN technique [70]. As main drawback here, the extracted segments often merge small dynamic traffic participants and large static environment parts. Therefore, such mixed regions were predicted falsely as unchanged regions, resulting in low recall values during the evaluation.

Performing the radius search at point level [68, 69] rather than across segments produced better results, however, this point level approach could not compensate for the irregular density characteristics of the RMB data. The voxel-based approach [72] outperformed both techniques regarding the F1-score metrics. However, all of them are not or just partially able to find pedestrians staying near to the static object (with a distance under voxel size, or under radius length, respectively), yielding many false-negative points. Conversely, the proposed RangeMRF model is able to distinguish the static and dynamic point regions, which difference is demonstrated by a few qualitative samples in Fig. 4.15 as well. By applying the method of [66], which compares points to nearest triangles, we could remarkably reduce the number of false-negative hits. However, building a surface from the neighbours of each point is a time-consuming step, the computation of one measurement frame takes here around 3-4 minutes using an Intel Core i7-7700K CPU@4.2GHz desktop

Method	Precision	Recall	F1-score	Computation time [s]
Segment-NN	0.8549	0.4834	0.6176	0.5649
Point-NN	**0.9763**	0.5982	0.7419	1.2934
Point-Triangle	0.9011	0.7993	0.8472	215.78
VOXEL	0.8017	0.7902	0.7959	2.5176
RangeMRF	0.8695	**0.8769**	**0.8732**	**0.1426**

Table 4.7: Comparative evaluation of various change detection methods and the proposed RangeMRF approach in cluttered areas for the Foreground change class (F).



(a) Camera images (for visual check) (b) Reference area in the MLS map (c) Voxel-based change detection [72] (d) Proposed change detection

Figure 4.15: Change detection results in crowded sidewalk areas, with the presence of many static and dynamic objects. Camera images of column (a) were taken synchronously with the RMB Lidar points clouds, but they are only used for visual verification. Purely Lidar-based detection results are shown in column (c) for a reference method, and in column (d) for the proposed method: red points correspond to dynamic change (vehicle, tram, pedestrians, further objects missing from the map: ticket station, benches), blue points present static environment parts. (Note: shading the red color in the second row is only applied for better visualization.)

computer.

The proposed RangeMRF model's segmentation step, which works in the range image domain of each frame, needs around 145 milliseconds, while the voxel-based approach takes around 2 (respectively 7) seconds for a given frame with parameter setting $w = 100$ cm (respectively $w = 10$ cm) using the same computer setup.

In summary, Fig. 4.11, Fig. 4.15, and Table 4.7 confirm the efficiency of the proposed RangeMRF method in real-world scenarios. We have quantitatively demonstrated its advantage versus the reference models, which produced either at least 7% lower F1-score rates on cluttered point clouds regions [68–70, 72], or they needed a significantly longer computational time [66].

4.4 Map-guided object-level scene analysis

Estimating the change mask is often a first step towards more ambitious goals of scene understanding, such as identifying and tracking all dynamic objects in the environment. In this section, we demonstrate that using the proposed cross-source RMB-MLS registration and change detection algorithms, we can achieve a notable performance improvement even for a state-of-the-art Lidar-based object detector [4, 5]. Comparative tests are provided in high traffic road sections of the *SZTAKIBudapest* Benchmark, and we achieved an advantage of 5.96% in precision, 9.21% in recall and 7.93% in F1-score metrics compared to the state of the art.

4.4.1 Related work in Lidar-based object detection

During the past few years, many geometric [52] and deep learning [51,74–77] based algorithms appeared in the literature for dynamic object detection. They operate on raw RMB Lidar frames and provide as output sets of oriented bounding boxes for various dynamic object categories such as vehicles, pedestrians or bicycles. However, due to the sparseness of the RMB Lidar measurements, there are a number of limitations of these approaches, especially in complex, crowded scenarios with many traffic participants.

On one hand, many *false positive* hits can be detected in point cloud regions containing static scene objects with similar appearance and context parameters to the dynamic target objects (see Fig. 4.16). On the other hand, the point cloud blobs of several dynamic objects can be occluded or merged with static street furniture elements, hiding them from the attention of the detector, which may cause many *false negative* predictions.

Exploiting prior information from city maps for improved object detection is a quite new research area, with only a few related techniques in the literature [78, 79]. The HDNET [78] approach uses a prior *road map* with local *ground-*

Figure 4.16: Falsely detected object samples overlapping with the background map. Color codes: predicted vehicle (red box), predicted pedestrian (blue box).

height data as reference, which helps in eliminating false object candidates detected out of the road, or above/under the ground level. The MapFusion [79] method extract features from three HD map layers (driveable areas, walkway, parking area) to exploit information about the structure of the roads. However, none of them deal with the confusion of dynamic objects with static entities from the map, and therefore they cannot adjust the missing object rate.

Our goal is to overcome this limitation, so we proposed a novel method to directly exploit 3D information from the scene.

4.4.2 Proposed method

We propose a method that takes a sparse RMB Lidar frame and the reference area of a 3D point cloud map as input and outputs the dynamic objects of the scene. Initially, we apply a traditional Lidar-based object detection algorithm to predict a set of object candidates in the current RMB Lidar frame. As a basis of comparison, we have chosen the *PointPillars* [51] state-of-the-art object detection method, which can predict object-candidates from multiple classes, together with their 3D oriented bounding boxes and class confidence values.

To refine the output of the object detection, we accurately register the input RMB Lidar point cloud to the prior MLS map by the cross-source point cloud registration algorithm introduced previously in Sec. 4.1. After the alignment, we apply a map-based, probabilistic validation step against the MLS model [4], to remove *false positive* object predictions, such as a vehicle de-

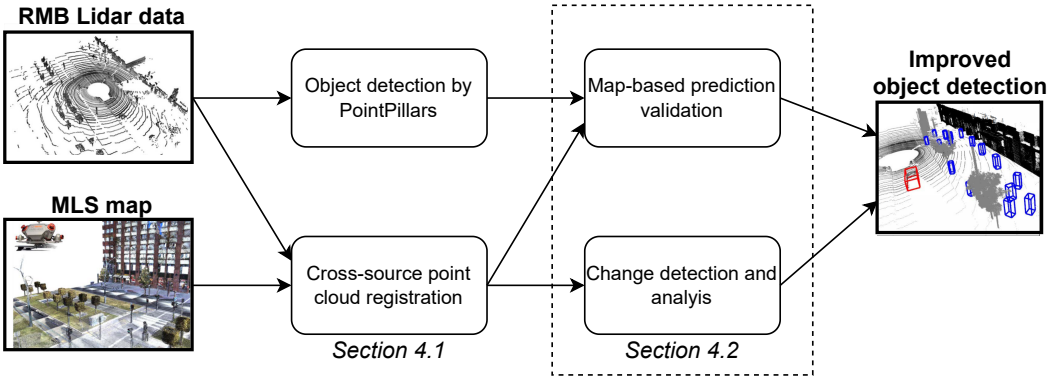


Figure 4.17: Workflow of the proposed approach.

tected in the pedestrian area of a tram stop. This step will be introduced in Sec. 4.4.2.1. Next, to eliminate the *false negatives*, we subtract the MLS map and the already detected dynamic objects from the actual Lidar frame using the change detection technique proposed in Sec. 4.3. Finally, we extract object candidate blobs in the remaining dynamic regions, and we attempt to identify these *previously undetected* dynamic objects by a Support Vector Machine (SVM)-based blob classifier. This step is described in Sec. 4.4.2.2. A high-level overview of the proposed method is displayed in Fig. 4.17.

4.4.2.1 False positive object removal by map-based validation

False positive objects often overlap with static obstacles of the background scene, thus they can be identified through analyzing their location in the registered 3D map. We propose a 2D probabilistic approach to manage this problem whose main steps are summarized in Fig. 4.18. First, taking a top-view analysis, we project both the RMB Lidar and the registered map point clouds to a discrete grid on the ground plane, with a resolution of 10 cm. Thereafter, we assign to each (i, j) cell two competing potentials describing the foreground ($P_{\text{fg}}(i, j)$) and background likelihoods ($P_{\text{bg}}(i, j)$). Foreground values are determined by the object detection output: for each cell covered by an object candidate, we take $P_{\text{fg}}(i, j) \in [0, 1]$ as the prediction score (i.e., confidence value) of the object detection network regarding the given object. The remaining cells receive $P_{\text{fg}}(i, j) = 0$. On the other hand, the background likelihoods are calculated from the projected MLS point cloud. If cell (i, j) is

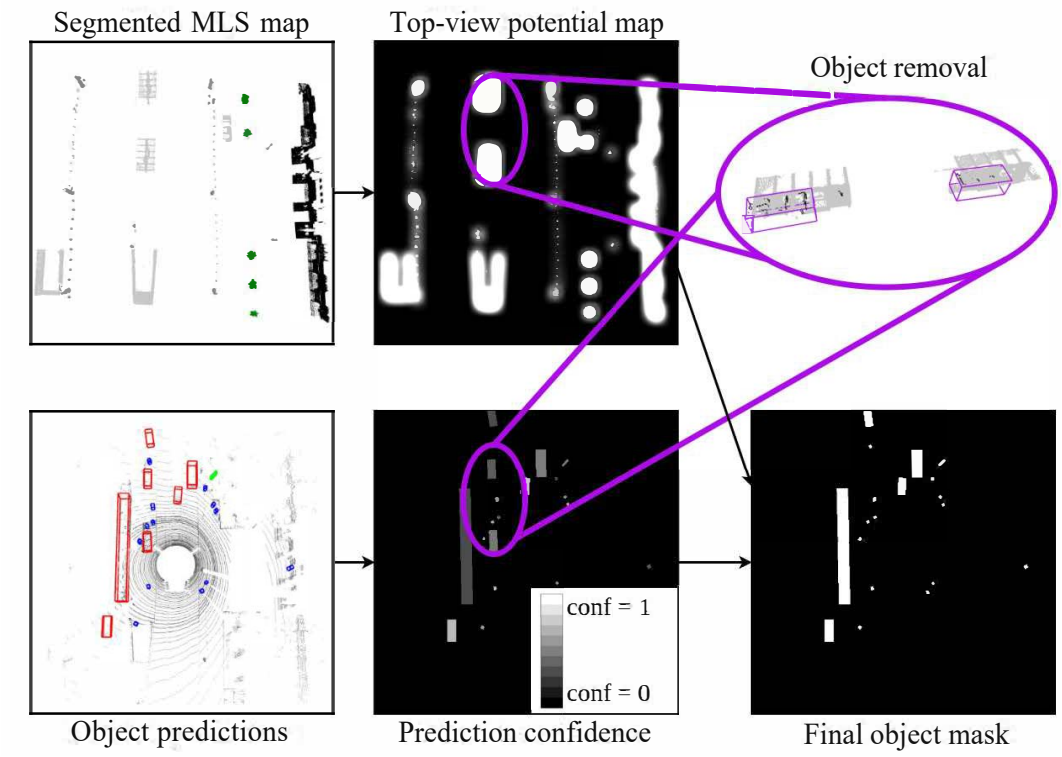


Figure 4.18: Overview of the proposed false positive object removal step.

occluded by a static obstacle in the MLS map, we set $P_{\text{bg}}(i,j) = 1$, while for cells near to the boundaries of static objects we use a distance-based Gaussian attenuation in the P_{bg} until 1 meter in any directions (with variance parameter $\sigma = 10$). For the remaining cells, we set $P_{\text{bg}}(i,j) = 0$.

Using the constructed likelihood maps, we remove all object candidates, which cover any cell (i,j), where $P_{\text{bg}}(i,j) \geq P_{\text{fg}}(i,j)$. Note that the adopted Gaussian soft boundary also ensures robustness of the approach against small registration errors.

4.4.2.2 Search for missing objects via change detection

On the other hand, for reducing the number of false negative objects, we subtract the MLS point cloud map from the actual Lidar frame using our RangeMRF change detection algorithm (Sec. 4.3). In this way, many previously undetected dynamic objects can be distinguished from the static scene elements that closely surround them (e.g. pedestrians from the bus stop, see also Fig. 4.15(a)), and then they can be separated by a region growing algo-

No.	Description	Dim
f_1	Number of points included in the object	1
f_2	The minimum distance to the object center	3
f_3	3D covariance matrix of the object points	6
f_4	Principal component of the object	3
f_5	3D bounding box sizes (height, width, depth)	3

Table 4.8: Feature vector used for SVM classification

rithm [52]. Finally, we identify the previously undetected objects of interest by a Support Vector Machine [80] based blob-classifier [4], which classifies the blobs based on the set of features listed in Table 4.8. After classification, the blobs labeled as vehicles or pedestrians are added to the list of detected objects.

4.4.3 Evaluation

As a baseline method, we have chosen the PointPillars [51] state-of-the-art object detection method. We have trained PointPillars on the KITTI [8] 3D object detection benchmark and some annotated samples from our *SZTAKIBudapest* dataset.

For quantitative evaluation, we have selected five heavy traffic road sections from the *SZTAKIBudapest* Benchmark. From each location, the evaluation dataset contains 50 different frames; and in average 5 vehicles and 16 pedestrians are present in a single time frame. The numerical performance results compared to the original *PointPillars* [51] output are summarized in Table 4.9. By combining our proposed change detection model with the selected

	Class	Precision	Recall	F1-score
PointPillars [51] baseline	Pedestrian	**0.9562**	0.6742	0.7908
	Vehicle	0.7519	0.8819	0.8111
	Both	0.8875	0.7222	0.7964
PointPillars [51] with our proposed map-based improvements	Pedestrian	0.9460	**0.8443**	**0.8752**
	Vehicle	**0.9502**	0.8819	**0.9138**
	Both	**0.9471**	**0.8143**	**0.8757**

Table 4.9: Object-level evaluation with the PointPillars [51]

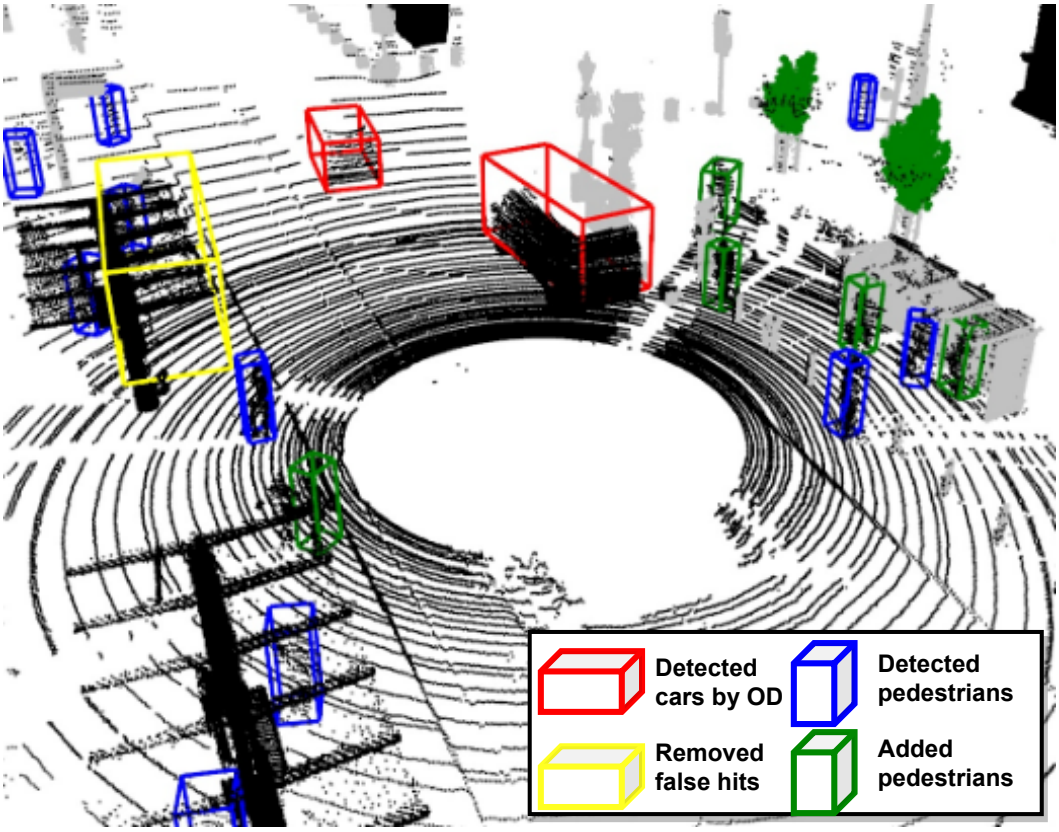


Figure 4.19: Improving PointPillars [51] based object detection (OD) in a complex scene. The correct initial object predictions are marked by red and blue for cars and pedestrians, respectively. Yellow box marks a falsely detected object, which was removed by us using map information, green boxes show pedestrians erroneously ignored by OD, but found after using our change detection approach.

object detection approach, we are able to significantly increase the number of recognized pedestrians and decrease the number of falsely predicted vehicles, improving the F1-score for each class with around 10%.

Fig. 4.19 displays a qualitative comparison of PointPillars alone and with our proposed correction method. In this scene, the proposed model provides us a comprehensive scene interpretation, although several vehicles and pedestrians are jointly present.

4.5 Implementation details and sample codes

All the developed algorithms for point cloud handling and processing were implemented in the C++ [108] programming language with the OpenCV [109]

and PCL [110] libraries. The reference PointPillars [51] neural network model was implemented and trained in the Python [111] programming language with the Pytorch [112] framework.

At the time of publication, the registration and change detection datasets and sample codes were publicly available at the following link: www.github.com/sztaki-geocomp/Lidar-SCU.

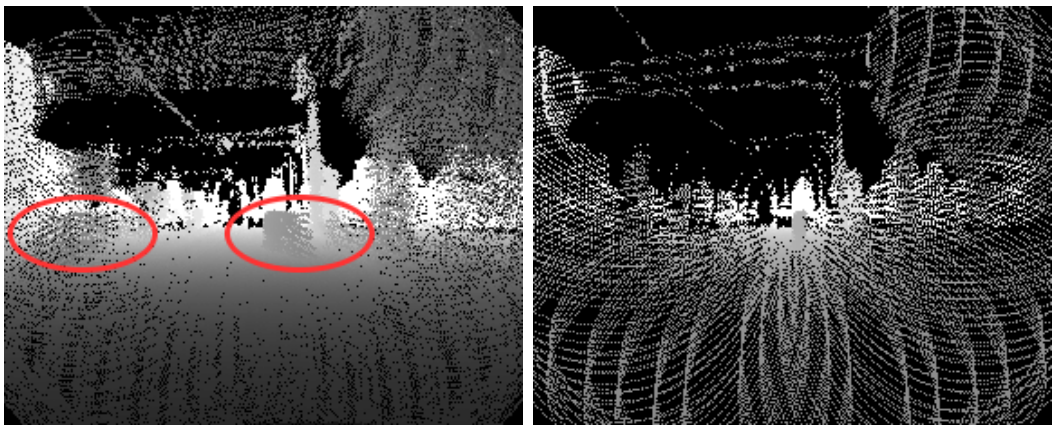
4.6 Conclusion of the chapter

This chapter presented a novel method for urban scene analysis between point clouds with significantly different density characteristics. The proposed algorithms can be used to localize sparse and instantly sensed point cloud data captured by AVs in a high-density 3D point cloud map with an accuracy of up to a few centimeters in median point distance. We also proposed a robust novel pose estimation and tracking algorithm, which operates with 20-25 fps by combining the introduced registration technique with a Kalman-filter. Based on the registered point clouds, we proposed a Markov Random Field model that can separate different sources of changes with an overall F1-score of 92% for complete RMB point cloud frames and 87% for complex sidewalk areas. We provided several experiments to show the advantages of the new methods versus the state of the art, and we also introduced an ADAS example as a possible efficient industrial utilization of the change detection results for dynamic object detection. We estimate that with a parallel implementation of many substeps, the whole workflow can run with around 5-10 fps.

Chapter 5

Real-time densification of sparse Lidar data

Lidar-based range measurements are widely represented by depth images [6], where, as the main advantage, they enable to adopt 2D convolution operations and effective image-based neural network architectures [81,82] during data processing. However, the data captured by Lidars is often very sparse, while its characteristics may vary depending on the sensors' scanning technology which results sparse and incomplete depth images which are challenging to interpret. In this context, *depth completion* algorithms focus on the problem to estimate



(a) Large integration window ($t_\Delta = 1$ s) results in blurred dynamic objects

(b) Narrow integration window ($t_\Delta = 200$ ms) results in a too sparse depth map

Figure 5.1: A dynamic scene captured by a NRCS Lidar with different t_Δ integration windows. A large integration time (a) induces several blurring artifacts, while a narrow integration window (b) yields the loss of details. Blurred pedestrians are marked by red ellipses.

dense depth images from the Lidar-acquired sparse range data.

In this chapter, we propose a novel depth completion algorithm, which utilizes measurements of a non-repetitive circular scanning Lidar. A special challenge for NRCS Lidars is to efficiently balance between the spatial and the temporal resolution of the recorded range data using a suitable integration window [22].

On one hand, as shown in Fig. 5.1(a), allowing larger integration time ($t_\Delta > 1$ s), the laser beams cover a higher proportion (around 90%) of the FoV yielding high spatial measurement resolution. However, the potential ego-motion of the Lidar's platform (e.g., vehicle or robot) and the dynamic objects in the surrounding area induce various artifacts, such as blurred shapes of the observed vehicles, pedestrians or buildings, which phenomena complicate dynamic event analysis. On the other hand, if the measurements are collected within a narrow time window (e.g., in 200 ms) they are spatially more precise, however, the resulting point clouds are notably sparse (around $48k$ points, up to 40% FoV coverage), which fact yields a significant loss of details across the spatial dimension of the FoV (see Fig. 5.1(b)).

In this chapter, we aim to overcome the above-mentioned challenges caused by the spatio-temporal trade-off of the NRCS Lidar based perception, and propose a novel deep learning based approach for densifying sparse NRCS Lidar data while keeping its spatial accuracy high. As the main contribution of the chapter, we propose a novel deep neural network called *ST-DepthNet*, which extends the classical U-Net architecture with a spatio-temporal encoder branch for utilizing consecutive sparse measurements captured by NRCS Lidars. Our model produces spatially precise high-density depth data using a spatial decoder branch following effective temporal pooling steps. We qualitatively and quantitatively evaluate the proposed algorithm on our constructed *LivoxCARLA* and *LivoxBudapest* datasets, and experimentally demonstrate its advantages against two state-of-the-art reference methods.

5.1 Outline of the proposed method

Our proposed spatio-temporal (ST) deep neural network called *ST-DepthNet* (Fig. 5.2) operates in the range domain and expects as input multiple sparse depth maps captured consecutively in time by a NRCS Lidar equipped

on a moving platform, with using a narrow (i.e., 200 ms) integration window for each frame. As output, the network provides a dense, high-quality range image of the same FoV, which does not reflect the sensor's original scanning artifacts (i.e., visible trails of the circular scanning pattern). The architecture of *ST-DepthNet* was directly designed to exploit both spatial and temporal patterns in the input NRCS data for depth completion, by extending a U-Net-like architecture [81, 83] with Conv2DLSTM [82] layers. Conv2DLSTM layers [82] are often used for tasks involving sequential data that also have a spatial component (e.g., video processing). They combine a two-dimensional convolutional (Conv2D) and a Long Short-Term Memory (LSTM) layer [84] to process both spatial and sequential information simultaneously. Therefore, they allow the models to learn spatial features from each frame using Conv2D operations while considering the temporal context and dependencies between frames using LSTM-like mechanisms.

5.2 Related work in depth completion

In this section, we present a study of the state-of-the-art depth completion techniques and challenges. In the past few years, research on Lidar-based approaches emerged as a hot topic in the literature, due to the availability of popular public datasets like the KITTI Depth Completion Benchmark [85], which contains over 93 thousand RGB images with the corresponding rotating multi-beam Lidar projected sparse depth measurements.

Therefore, the majority of the recent methods focus on completing depth maps obtained from RMB Lidars fused with camera images as guidance to recover the pixels with missing depth measurements [86, 87]. However, images may not provide eligible information in cases of sudden illumination changes [87] or in low-light environments [88]. In these cases, depth completion must be performed solely based on sparse Lidar range measurement samples, which includes significantly harder challenges [89].

First, without relying on external sources (e.g., high-resolution RGB images), edges and other finely textured structures on the generated depth images are often missing, blurred or distorted [89, 90]. In [89], global and local depth variations are separated based on the fact that in the wavelet representation of the images, the fine structures appear in the high-frequency domain while

the global regions are defined by the low-frequency coefficients. In order to exploit this phenomenon, they introduce a frequency-based recurrent depth coefficient refinement scheme. The difficulty of data upsampling near the edges also appears in the work of Savkin [90], where feature extraction by an edge convolution layer is used to strengthen the precision at fine 3D structures. In our approach, we recover the fine structures by adding an appropriate edge-loss term [91] to our loss function, instead of performing edge enhancement by a dedicated sub-network.

Second, a limitation of many existing depth completion methods is that they generate new range values for all image pixels, instead of filling only the missing information [27]. Therefore the Implicit Lidar Network by Kwon et al. [27] learns the weights of an interpolation function for 3D point cloud completion, thus the original measurements are not modified and only the missing points are estimated. For similar reasons, our solution connects the last sparse input image to the output by a direct skip connection to force our proposed model to keep the original sparse, but precise range map and complete the missing regions, instead of overwriting the whole input image with completely new values.

The most closely related methods to our approach that focus on Lidar-only depth completion are [92] which effectively combines morphological operations and bilateral filtering, and [88] that investigates different sampling strategies for training a generative adversarial network. However, as our experiments show in Sec. 5.4, both approaches are highly sensitive to the measurement characteristics of the applied Lidar sensor and fail to accurately compensate for the irregular, non-repetitive sampling pattern of NRCS Lidars. As NRCS Lidars are relatively new to the market, to the best of our knowledge, this thesis is the first to provide a dataset and method utilizing information for depth completion propagated from their measurements.

5.3 The proposed depth completion method

The goal of the proposed solution is to produce a high-quality, dense and spatially precise point cloud stream from measurements of a single NRCS Lidar sensor. Our approach consists of two main steps: First, the consecutive measurements of the NRCS Lidar are grouped to form discrete time frames, using

a narrow, 200 ms integration window (up to 40% FoV coverage in each frame). Thereafter, within a frame, the distances of the measured 3D field points from the sensor are assigned to corresponding pixels in a high-resolution range image. By each actual time frame, the last five collected depth images (covering together around 95% of the FoV) are fed to the *ST-DepthNet* depth completion network, which composes a high-quality range image as output, with almost 100% FoV coverage, also eliminating the motion blurring artifacts. The output high-quality range image can be backprojected to the 3D space as well.

5.3.1 Range image generation

In our approach, the captured sparse point clouds (collected within t_Δ = 200 ms) are converted from the Cartesian (x,y,z) to the spherical (distance, azimuth, elevation) polar coordinate system. Then, a 2D pixel lattice is generated by quantizing the horizontal (azimuth) and vertical (elevation) FoVs. In the resulting range images, the horizontal and vertical pixel coordinates represent the polar azimuth and elevation angles, while the pixel's depth value encodes the distance of the corresponding point.

In our experiments, we exploit the parameters of the Livox AVIA NRCS Lidar sensor [22]. The sensor's FoV is mapped onto a 400×400 pixel lattice, which resolution (5.6 px/°) yields both high spatial accuracy and reasonable computational requirements. As experienced, the density of the recorded valid range values is decreasing towards the peripheral regions of the range image due to the nature of the circular scanning technique: the scanning pattern crosses the optical center of the sensor significantly more frequently, than the FoV's perimeter, making the central regions of the range images densely filled, and leaving peripheral areas notably sparse (see Fig. 2.6 and 5.1). As a result of using an integration time window of 200 ms for collecting the consecutive time frames, around 60% of the range image pixels receive undefined range values. Such a level of sparseness of the range image makes it difficult to efficiently visualize the data or to perform scene analysis, emerging the need for the proposed depth estimation approach.

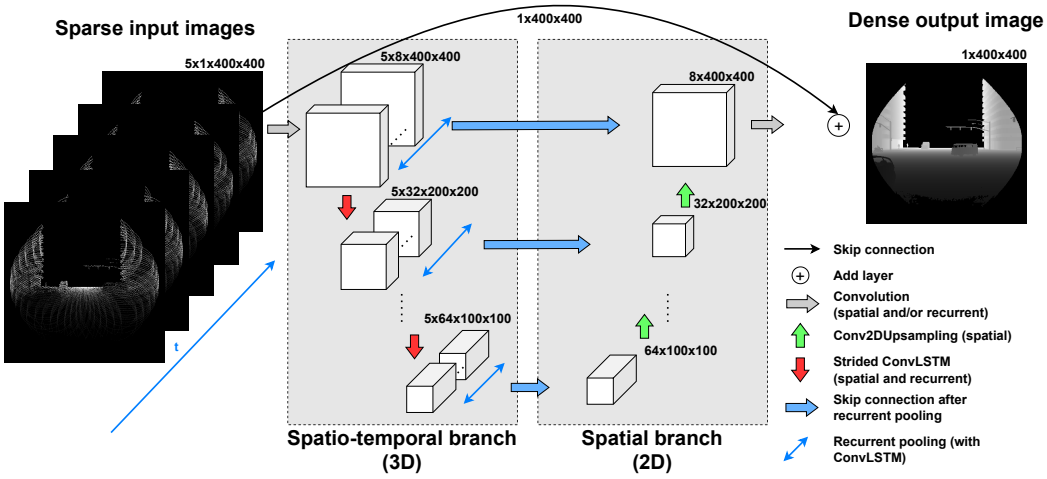


Figure 5.2: The architecture of the proposed ST-DepthNet network.

5.3.2 ST-DepthNet architecture

Next, we use a range image sequence acquired by the NRCS Lidar as input to the proposed *ST-DepthNet* deep neural network (Fig. 5.2). As discussed earlier, sparse measurement frames collected in 200 ms time windows cover only a low proportion of the defined 400×400 range image lattice. On the other hand, using a 1 s time frame, the collected point set covers almost fully the sensor's FoV (Fig. 2.6), but it is affected by motion blur. Nevertheless, we can expect that the measurements from the last 1 s time interval always contain dense range information from the scene. Thus to also prevent blurring, we take five consecutive ,,sparse" range images (each one recorded in 200 ms) as our network's input.

Since the main goal is to generate a high-quality output image from the sparse range image inputs, we have adopted an image-to-image U-Net [81] like architecture. The original U-Net neural network [81] is characterized by its U-shaped architecture, which consists of an encoder and a decoder part with skip connections. The encoder part typically contains several convolutional layers followed by pooling layers (e.g., max-pooling) that progressively reduce the spatial dimensions of the input image while learning hierarchical features. The decoder part consists of upsampling layers (e.g., transposed convolutions or bilinear upsampling) that gradually increase the spatial dimensions of the feature map. The decoder's goal is to produce an output image with the same

spatial dimensions as the input image. Meanwhile, at each hierarchical level, the skip connections connect the corresponding layers between the encoder and decoder and concatenate feature maps at each spatial resolution. This allows the network to preserve fine-grained details during the decoder process and helps in capturing both low-level and high-level features.

In the proposed model, we extended the encoder part of a U-Net network enabling to exploit temporal connections where the input is an image sequence, by utilizing Conv2DLSTM layers presented first in [82]. Let us introduce a regular Long-Short Term Memory (LSTM) cell, which has a memory state C_t and a final state H_t. At each timestep t, the memory is updated as a function of its current input X_t, previous final state H_{t-1} based on an input gate i_t, while the propagation of its previous value C_{t-1} depends on a forget gate f_t. The propagation of the memory state C_t to the final state H_t depends on the output gate o_t. In each dependency, from state α to β, there is a weight term $W_{\alpha\beta}$ and a bias term b_α. A Conv2DLSTM cell operates similarly to a regular LSTM cell, with an extension that the input X_t, memory state C_t and final state H_t with their respective gates (i_t,f_t,o_t) are 3D tensors – with one temporal and two spatial dimensions – and both the spatial and recurrent transformations are convolutional (marked by $*$) and not element-wise (marked by $\circ$), making it able to propagate spatio-temporal features:

$$
\begin{aligned}
C_t &= f_t \circ C_{t-1} + i_t \circ \tanh(W_{xc} * X_t + W_{hc} * H_{t-1} + b_c) \\
i_t &= \sigma(W_{xi} * X_t + W_{hi} * H_{t-1} + W_{ci} \circ C_{t-1} + b_i) \\
f_t &= \sigma(W_{xf} * X_t + W_{hf} * H_{t-1} + W_{cf} \circ C_{t-1} + b_f) \\
o_t &= \sigma(W_{xo} * X_t + W_{ho} * H_{t-1} + W_{co} \circ C_t + b_o) \\
H_t &= o_t \circ \tanh(C_t)
\end{aligned}
$$

Hence, in our proposed approach, we keep a spatio-temporal three-dimensional (two spatial and one temporal) encoder branch at the whole left side of the U-Net structure. On the other hand, the decoder branch of our proposed network is purely two-dimensional, in order to accurately restore the single output image of our interest. Skip connections at each level are performed by recurrent pooling utilizing the last output of a Conv2DLSTM layer which represents features of the last 200 ms measurement.

As an extra modification, we directly connect the last input image to our output. With this extension, which is also supported by our ablation experi-

ments (see Sec. 5.4), we can exploit that the last and most up-to-date input image contains spatially precise depth data, and therefore, the proposed network only has to learn the missing regions of the range image [27].

5.3.3 Training process

The proposed *ST-DepthNet* network is responsible for learning and predicting a high-density range image using a sparse input range image sequence. To deal with the challenging artifacts presented in Sec. 5.2, our *loss function* $\mathcal{L}$ is composed of three main components.

First, we calculate the L1 Loss ($\mathcal{L}_{\text{L1}}$) as the mean absolute error between the generated and the GT depth images to force detailed, pixel-level accurate predictions. Second, we adopt the structure similarity index measure ($\mathcal{L}_{\text{SSIM}}$) proposed by [93], which quantifies the perceived difference in luminance, contrast and structural information between the predicted and GT depth images using a variety of known properties of the human visual system. Third, we also utilize a smoothness or edge loss term ($\mathcal{L}_{\text{EDGE}}$) specifically proposed for depth images by [91], which induces sharp contours on the generated images, thus spatially precise boundaries are enforced between objects in the 3D space. Our final *loss function* can be expressed therefore as follows:

$$\mathcal{L} = \alpha_1 \mathcal{L}_{\text{SSIM}} + \alpha_2 \mathcal{L}_{\text{L1}} + \alpha_3 \mathcal{L}_{\text{EDGE}}.$$

Following a parameter optimization step (see Sec. 5.4), $\alpha_1 = 0.7$, $\alpha_2 = 1.4$ and $\alpha_3 = 1.5$ were used in the final model. The loss function was minimized by the Adam optimizer [94]. The learning rate was set to 0.0002 and the decay rate of the first moment to 0.5. We have trained our model on the *LivoxCARLA* dataset for 10 epochs which took around 27 hours on a NVidia GeForce GTX 1080 Ti graphical processing unit (GPU).

5.4 Experiments

We have quantitatively evaluated the proposed method using the *Livox-Carla* dataset, exploiting its sparse input–dense output range image pairs generated by a simulated car-mounted NRCS Lidar sensor during virtual drives in dense city environments with several dynamic traffic participants (humans, vehicles, bikes). Besides quantitative validation, we also evaluated qualitatively

the performance of the proposed method on real data using the *LivoxBudapest* dataset. Both datasets are introduced earlier in Sec. 3.2.

5.4.1 Evaluation metrics

During quantitative analysis, we performed evaluation in both 2D and 3D, analysing the generated range images, and the backprojected 3D point clouds, respectively.

5.4.1.1 2D errors

For measuring the similarity between the generated range images to the GT, we adopted the following metrics from the KITTI Depth Completion Benchmark [85]:

- RMSE: Root mean squared error [mm]

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(I_{P_i} - I_{GT_i}\right)^2}$$

- MAE: Mean absolute error [mm]

$$\text{MAE} = \sum_{i=1}^{N}\left|I_{P_i} - I_{GT_i}\right|$$

- iRMSE: RMSE of the inverse depth [1/km]

$$\text{iRMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{I_{P_i}} - \frac{1}{I_{GT_i}}\right)^2}$$

- iMAE: MAE of the inverse depth [1/km]

$$\text{iMAE} = \sum_{i=1}^{N}\left|\frac{1}{I_{P_i}} - \frac{1}{I_{GT_i}}\right|$$

In the above equations, I_{P_i} denotes the ith pixel of the image generated by the actual method, while I_{GT_i} is the ith pixel of the corresponding GT image. N denotes the number of pixels, in our case $N = 400 \times 400$. Using the two direct depth-based errors (RMSE, MAE), we can compare the absolute accuracy of depth estimates in meters. However, in some cases, for example the large errors for distant objects can disproportionately affect the overall error value using these metrics. On the other hand, the inverse depth-based errors (iRMSE, iMAE) focus on relative error improvements and may be more relevant in scenes with objects at varying distances. Motivated by this, in the upcoming experiments, we used both depth and inverse depth-based errors.

5.4.1.2 3D errors

Besides range image based evaluation, we also compared the generated point clouds to the reference model in the 3D space. Let us denote the GT and a predicted point cloud by P_{GT} and P_{P}, and the number of points in P_{GT} and P_{P} by $\#P_{\mathrm{GT}}$ and $\#P_{\mathrm{P}}$, respectively. We evaluate the quality of the predicted point cloud with respect to the GT data using the symmetric normalized Chamfer distance (NCD) and normalized median distance (NMD) [1], while these evaluation measures are also used to compare the performance of different baseline algorithms in the 3D space:

$$
\begin{aligned}
S_P &= \sum_{p \in P_{\mathrm{P}}} \min_{q \in P_{\mathrm{GT}}} ||p-q||^2 \\
S_{GT} &= \sum_{q \in P_{\mathrm{GT}}} \min_{p \in P_{\mathrm{P}}} ||p-q||^2 \\
M_P &= \operatorname*{Med}_{p \in P_{\mathrm{P}}} \min_{q \in P_{\mathrm{GT}}} \sqrt{||p-q||^2} \\
M_{GT} &= \operatorname*{Med}_{q \in P_{\mathrm{GT}}} \min_{p \in P_{\mathrm{P}}} \sqrt{||p-q||^2} \\
Q_{\mathrm{NCD}}(P_{\mathrm{P}}, P_{\mathrm{GT}}) &= \sqrt{\frac{1}{2}\Big(\frac{S_P}{\#P_P} + \frac{S_{GT}}{\#P_{GT}}\Big)} \\
Q_{\mathrm{NMD}}(P_{\mathrm{P}}, P_{\mathrm{GT}}) &= \frac{1}{2}\Big(M_P + M_{GT}\Big)
\end{aligned}
$$

5.4.2 Ablation study and hyperparameters

For optimizing the network structure, we investigated the effect of how deep the proposed model integrates temporal information in the network architecture. In the first setup (No fusion), we trained the network without utilizing temporal data and considering only the measurements from the last 200 ms. In the second setup (Early fusion), we fused the multitemporal information only in the first Conv2DLSTM layer and the remaining layers remained pure spatial convolutions. Finally, as proposed, we propagated the temporal information through the whole feature downscaling branch (Late fusion). Furthermore, in each setup, we examined the effect of including/excluding U-Net-like skip connections in the network (Inner levels) and to directly bind the last input and the output depth image (Output). According to our comparative results displayed in Table 5.1, the proposed late fusion approach produced the small-

Temporal fusion	Skip connections	RMSE ↓	MAE ↓
X	X	3512.95	1549.36
X	Inner levels	3367.75	1353.49
X	Inner levels+Output	2869.84	1383.10
Early	X	2969.22	883.92
Early	Inner levels	2767.87	832.87
Early	Inner levels+Output	2129.39	686.34
Late	X	2672.20	800.16
Late	Inner levels	1897.99	523.57
Late	**Inner levels+Output**	**1799.16**	**440.42**

Table 5.1: An ablation study of the proposed ST-DepthNet architecture

est RMSE and MAE values, while allowing direct skip connections between the latest sparse input frame and the predicted output significantly improved on the results at each temporal setup. Using these connections, the network learns to *complete* the missing regions of the sensor's sparse range map, while keeping high fidelity to the accurate range measurements from the last 200 ms time frame.

Next, we also performed hyperparameter optimization steps in the final late fusion based model where we compared different weight combinations of the $\mathcal{L}$ *loss function*'s subterms. The most relevant configurations are summarized in Table 5.2. First, using a relatively higher weight for the $\mathcal{L}_{\text{SSIM}}$ loss term results in smoothed edges and blurred fine structures and therefore it produces higher RMSE and MAE errors. On the other hand, if the weight of $\mathcal{L}_{\text{SSIM}}$ is significantly smaller than the weight of $\mathcal{L}_{\text{L1}}$, image regions with uniform depth remain noisy, resulting again in higher RMSE and MAE rates. As a good balance, we experienced that an optimal ratio between the weights of $\mathcal{L}_{\text{SSIM}}$ and $\mathcal{L}_{\text{L1}}$ is around $1:2$. Second, based on a dozen experiments, the point level $\mathcal{L}_{\text{L1}}$ and edge based $\mathcal{L}_{\text{EDGE}}$ loss terms are in the best balance with a weight ratio of around $1:1$.

Output skip layer	α_1	α_2	α_3	RMSE ↓	MAE ↓
X	0.85	1.00	0.90	4267.88	1494.07
X	0.85	1.00	1.00	3871.45	1366.16
X	0.60	2.50	2.50	2938.23	1512.53
X	0.70	1.00	1.20	2000.33	541.06
X	0.70	1.40	1.50	1897.99	523.57
✓	**0.70**	**1.40**	**1.50**	**1799.16**	**440.42**

Table 5.2: Different hyperparameter setups for the final model

Please note that although we initially performed multiple trainings with each parameter and network architecture setup, we did not experience significant differences between the results of the different runs. These strong trends made unnecessary the need for multiple runs at every parameter setup. Therefore, Table 5.1 and Table 5.2 display the results of single training runs.

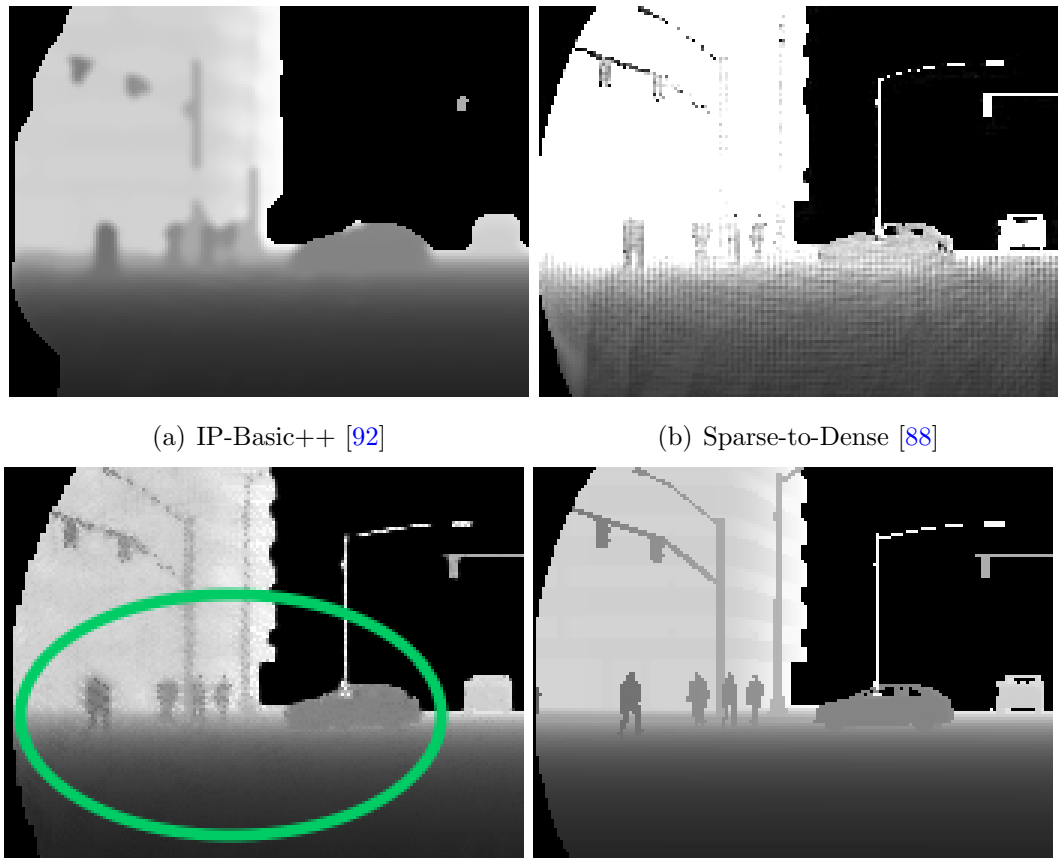
5.4.3 Reference methods

We have compared the results of the proposed *ST-DepthNet* model to related approaches published in the recent years. Note that the majority of existing methods [85–87] relies on fused Lidar based sparse depth maps and dense RGB images, therefore we cannot directly compare the proposed method to them, as we address Lidar-only scenarios. As the first baseline for comparison, we investigated how the sensor itself can produce high density images, by allowing a large integration window ($t_\Delta = 1$ s) to cover a high proportion ($> 95\%$) of the FoV. We refer to this method from now on as *Large integration*. As the second reference, we adopted an improved version of the method presented in [92], called hereafter as *IP-Basic++*, by optimizing its morphological operations to our irregular NRCS data and extending it with bilateral blurring. We have chosen as the third reference the approach of [88], called henceforward *Sparse-to-Dense*, which is proposed directly for Lidar-only perception, and we trained it on our *LivoxCarla* dataset, with the parameters described in [88]. To adopt the latter method to our dataset, we changed the size of the input layer from 480×480 to our range image lattice 400×400.

5.4.4 Comparative results

Next, we compare the *ST-DepthNet* to the above three reference methods on the *LivoxCarla* test set, in both 2D range image based and 3D point cloud based representations.

The overall mean values of the calculated 2D error rates are displayed in Table 5.3. Regarding all numerical quality measures, the performances of the *Large integration* and the *Sparse-to-Dense* [88] approaches are quite similar, the *IP-Basic++* [92] works better in average, while the proposed *ST-DepthNet* significantly outperforms all of them, reducing their RMSE errors by more than 1 meter.



(a) IP-Basic++ [92] (b) Sparse-to-Dense [88]

(c) Proposed ST-DepthNet (d) Ground truth

Figure 5.3: Fine structures (marked by green ellipses) recognized by the proposed ST-DepthNet approach, but remained partly or fully unrecognized (merged to wall or background) by the reference IP-Basic++ [92] and Sparse-to-Dense [88] approaches with respect to the ground truth data.

First, we can observe that the main sources for large errors of the *Large integration* method are the movement of the capturing platform and the presence of dynamic objects of the scene. Second, the *IP-Basic++* [92] approach predicts missing depth values more robustly on large, homogeneous surfaces,

Method	iRMSE↓	iMAE↓	RMSE↓	MAE↓
Large integration	74.745	21.12	4259.83	1119.41
IP-Basic++ [92]	170.02	24.31	2918.33	574.99
Sparse-to-Dense [88]	493.65	151.55	4583.75	1672.79
ST-DepthNet	**59.46**	**15.94**	**1799.16**	**440.42**

Table 5.3: Comparative results between 2D range images

Method	Q_{NMD}[mm]↓	Q_{NCD}[mm]↓
Large integration	1754.12	3830.62
IP-Basic++ [92]	1072.60	2241.69
Sparse-to-Dense [88]	4466.14	6065.44
ST-DepthNet	**687.36**	**1718.53**

Table 5.4: Comparative results in the 3D space

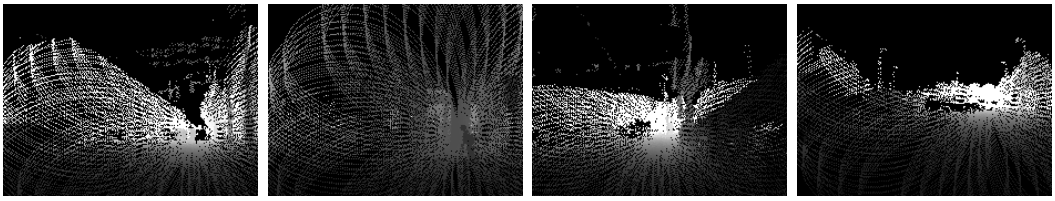
but fails estimating the fine details. Third, the depth image estimation by the *Sparse-to-Dense* [88] method keeps the trails of the circular scanning pattern of the NRCS sensor still visible, while scene objects and finely textured regions are often merged with their background. Fig. 5.3 demonstrates these limitations of [88] and [92] on a range image sample, where the proposed method performs significantly better.

We conducted further analysis in the 3D domain, by comparing the 3D ground truth scene models to point clouds backprojected from the range images generated by the proposed and reference methods. Errors obtained by calculating the symmetric Normalized Chamfer and Median Distance metrics are displayed in Table 5.4. As shown, the error of the proposed method is the smallest, by a margin of around a half meter regarding both metrics. Note that, while *Large integration* seems to work better than *Sparse-to-Dense* in 3D, this observation is mainly the consequence of the fact that object regions affected by motion blur can still have points close to GT in the 3D space, and vice versa.

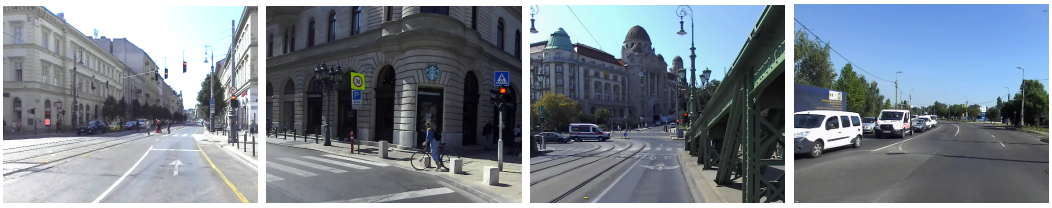
5.4.5 Analysis on real measurements

Beyond a comprehensive numerical evaluation on our synthetic *LivoxCarla* dataset, we also validated the proposed method on real Lidar measurement sequences of the *LivoxBudapest* set, supporting its future real-life application.

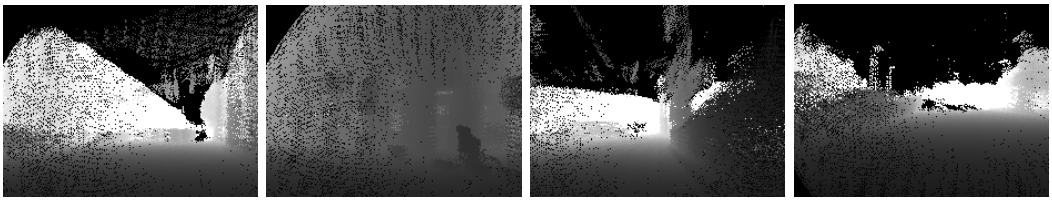
The *LivoxBudapest* test set contains three different scenarios: two pathway recordings from the city center (a *boulevard* and a *narrow street*), both around 1 km long, and a *speedway* section near the city, recorded on a path of around 3.5 km. Fig. 5.4 displays selected relevant sample frames from the three scenarios. We can observe that similarly to the experiments with synthetic data, the *IP-Basic++* [92] approach robustly completes missing values in case of larger surfaces (walls, ground areas and even vehicles), but fails to



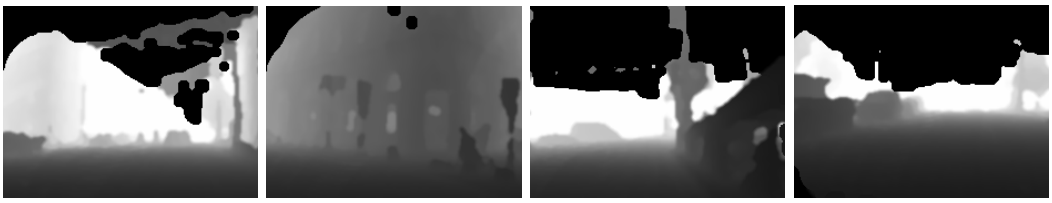
(a) Sparse input data captured in a 200 ms time window



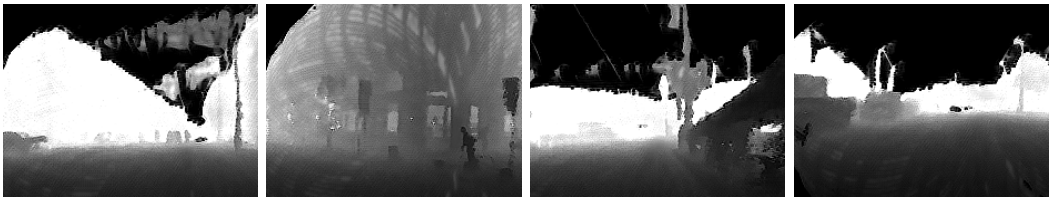
(b) RGB image for visual reference only



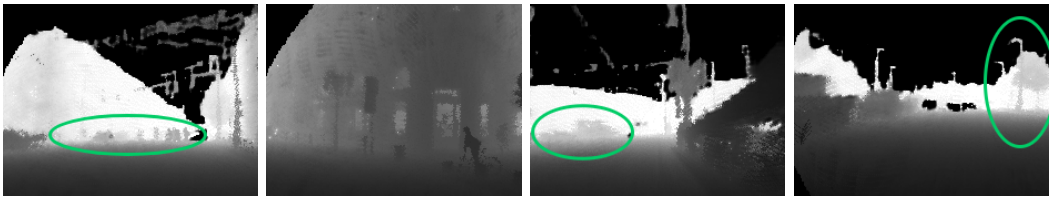
(c) Large integration time ($t_\Delta = 1$ s)



(d) IP-Basic++ [92]



e) Sparse-to-Dense [88]



(f) Proposed ST-DepthNet

Figure 5.4: Results on real measurements from the LivoxBudapest test set. Accurately predicted fine object structures by ST-DepthNet are highlighted with green ellipses.

Method/ score↑	Boulevard	Narrow st.	Speedway	Mean
Sparse input data	2.95	2.75	2.60	2.76
Large integration	4.30	4.45	3.75	4.17
IP-Basic++ [92]	5.75	5.65	5.60	5.67
Sparse-to-Dense [88]	6.10	6.35	6.40	6.28
ST-DepthNet	**7.15**	**7.20**	**6.65**	**7.00**

Table 5.5: Comparative survey results on real measurements

accurately estimate fine structures. For example, pedestrians in the first column of Fig. 5.4(d) are blurred into one object, while traffic lights and signs are partly merged to the background in the second and fourth columns of Fig. 5.4(d). The *Sparse-to-Dense* method cannot eliminate the rosetta patterns of the input Lidar measurements, which are typically visible on ground and wall areas (e.g., second column in Fig. 5.4(e)). The tendency of merging fine structures into larger surfaces is also notable: In the first and third columns of Fig. 5.4(e), vehicles and pedestrians are falsely merged to the wall behind them. Such artefacts can mean critical problems for urban scene understanding tasks, while as shown, they are handled better by the proposed *ST-DepthNet* approach (see regions marked by green). The *Sparse-to-Dense* method heavily blurs other fine structures (columns, traffic signs and lights, etc.) as well, as displayed in Fig. 5.4(e). Moreover, while objects close to the sensor are usually well recognizable for the human eye, they are often predicted at inaccurate distances with this method (e.g., cyclist in the second column of Fig. 5.4(e)). As for the *Large integration* method, it performs significantly worse on real data than on the simulated samples: its generated range images are extremely noisy, and if the platform is moving, structures are barely recognizable. Note that by large integration time the regions of moving street objects become blurred even if the Lidar platform is static.

Besides the above qualitative analysis, we conducted a survey for visual verification of the generated depth image streams, by asking 20 computer vision related experts to rate the quality of the input and the output of each method in all three videos with scores between 1 and 9, where 9 is the best possible score. The results provided in Table 5.5 confirm that the test subjects found the proposed method significantly better than the reference techniques.

In summary, the proposed *ST-DepthNet* method can better compensate for both the noisiness and the sampling pattern of the sensor data, while the

predicted distance values of dynamic objects or background scene structures are more accurate than in the depth maps of the reference approaches.

Regarding the computation time, for the prediction of a single for 400 × 400 depth image frame, the *ST-DepthNet* method needs 100 ms on an Intel Core i7-7700K CPU@4.2GHz desktop computer utilizing an NVidia GeForce GTX 1080 Ti GPU. The whole workflow (depth image generation, completion and visualization of the results) runs with 5 fps on an AMD Ryzen 7-6800H CPU@(8x3.2GHz) laptop computer utilizing an NVidia Geforce RTX 3070 GPU with 8GB memory, which enables real-time processing, since we collect each depth image within 200 ms.

5.5 Implementation details and sample codes

The process of generating depth images from point cloud scans was implemented in the C++ [108] programming language using the OpenCV [109] and PCL [110] libraries. The proposed *ST-DepthNet* neural network model was implemented and trained in the Python [111] programming language with the Keras [113] framework. The real-time demonstration of the whole workflow was implemented in the Robot Operating System [114].

At the time of publication, the depth completion datasets and sample codes were publicly available at the following link: www.github.com/sztaki-geocomp/ST-DepthNet.

5.6 Conclusion of the chapter

In this chapter a novel depth completion method called *ST-DepthNet* was proposed, which is capable of creating high-density depth images from sparse consecutive depth maps acquired by a NRCS Lidar. For training and quantitative evaluation, we constructed a new synthetic Benchmark set called *Livox-Carla*, and we shown that our approach outperforms two state-of-the-art reference methods. The usability of the proposed method on real NRCS measurement data has also been demonstrated using our recorded *LivoxBudapest* real-life dataset. In the future, the aim is to use the proposed method in intelligent robot and vehicle platforms, for improving the limited spatial resolution of NRCS Lidars.

Chapter 6

Conclusion and Outlook

6.1 Summary of contributions

This thesis presented novel methods for real-time environment analysis using low-resolution Lidar point cloud measurements.

In Chapter 4, we have presented a new method for urban scene analysis, which comprises 3D point cloud registration and change detection through fusing Lidar point clouds with significantly different density characteristics. The introduced method is able to extract dynamic scene segments (traffic participants or urban renewal regions) and seasonal changes (vegetation regions) from real-time 3D measurements captured by a Rotating Multi-beam (RMB) Lidar sensor mounted onto the top of a moving vehicle. As reference data, we relied on a dense point cloud-based environment model provided by Mobile Laser Scanning (MLS) systems. The proposed approach is composed of multiple novel steps. First, a novel cross-source point cloud registration algorithm was introduced, which can improve the alignment of the sparse RMB measurements to the dense MLS data, where conventional point-level registration or keypoint/segment matching strategies fail. Second, related to this, a real-time pose tracking approach was presented by integrating the registration results in a Kalman-filter-based dynamic vehicle model. Third, an efficient Markov Random Field-based change extraction step was implemented between the registered point clouds, which exploits the fact that due to geometric considerations of mapping with the given sensor configuration, the essence of the problem can be solved quickly in the 2D range image domain without information loss. Finally, an efficient utilization of the proposed change detection

approach was demonstrated for enhancing the performance of state-of-the-art Lidar-based object detection. Experimental evaluations were conducted on a new Benchmark set that contains three different heavy traffic road sections in city center areas covering in total nearly 1 km long pathway sections. Using the new Benchmark, we have quantitatively demonstrated the clear advantages of the new method against various reference techniques.

In Chapter 5, we introduced a novel depth image completion technique based on sparse consecutive measurements of a non-repetitive circular scanning (NRCS) Lidar, demonstrating the capabilities of a new, compact, and accessible sensor technology for dense range mapping of highly dynamic scenes. The proposed deep neural network called *ST-DepthNet* is composed of a spatio-temporally (ST) extended U-Net architecture, which takes a sparse range data sequence as input and produces a dense depth image stream of the same field-of-view ensuring a high level of spatial details and accuracy. For evaluation, we have constructed a new urban dataset, that – to our best knowledge as the first open Benchmark in this field – comprises various simulated and real-world NRCS Lidar data samples, allowing us to simultaneously train our model on synthetic data with ground truth, and to validate the result via real NRCS Lidar measurements. Using this new dataset, we have shown the superiority of our method against a densified depth map obtained from the raw sensor stream, and against two independent state-of-the-art Lidar-only depth completion methods [88, 92].

6.2 Open problems and future research

With our original goals and our accomplished contributions in mind, we now mention some open challenges and provide three different directions for interesting future research.

6.2.1 Automatic HD map generation from raw MLS point cloud measurements

In industrial practice, vector-based high-definition (HD) maps are often adopted to augment the AVs' limited RMB measurements with prior beliefs [15], as they store precise and high-quality metadata about the static parts of the environment [16]. Nevertheless, the creation and maintenance of

these HD maps need continuous attendance of human operators [95], moreover, the vectorization process may also drop out relevant point-level details. Following a different approach, in Chapter 4 of this dissertation, we utilized offline integrated Mobile Laser Scanning (MLS) measurements for this purpose. However, these point clouds could also be used to automatically create highly detailed vector maps [96]. This process however induces several challenges.

First, to create vector maps, the raw Lidar point cloud data needs to be segmented and classified into different objects like roads, buildings, vegetation areas, and obstacles. Recent deep learning-based point cloud classification approaches such as the PointNet++ [97] or SPLATNet [98] offer promising ways to automate this process on various sorts of artificial or real scanning-based point clouds. However, dealing with urban MLS data a number of particular challenges appear – such as the phantom effect caused by independent object motions – , while the raw Lidar measurements are typically noisy due to various factors like rain or dust as well. The work of Nagy et al. [19] addressed exactly these challenges and produced promising results.

Second, assuming accurate and proper segmentation and classification of objects in the MLS cloud, extracting 3D models from the structures are also challenging, especially for objects with complex shapes and features. Moreover, detecting and representing vegetation and overhead structures like trees, bridges, and power lines can be complicated [99], as they often have irregular shapes. In the literature, recent methods [96, 100] therefore focus on the automatic generation of driving lines or lanes from the MLS point clouds and do not deal with 3D obstacles or buildings in the environment. An automatic vectorization of these 3D structures is still highly challenging and it would be a great contribution to this field.

Third, HD maps need to be updated regularly to reflect changes in the environment. Automating the update process to account for new construction, road changes, or other alterations is also a non-trivial task while it could spare a lot of money and manual work for the industry. It would be an interesting and challenging task to investigate how the output of the proposed change detection algorithm could contribute to automatically updating the MLS point cloud maps.

6.2.2 Dynamic object detection by fusing semantic layer and obstacle information

In Sec. 4.4, we introduced a workflow to demonstrate that using the output of the proposed cross-source registration and change detection method, we can improve even on the state-of-the-art PointPillars [51] object detector using simple, explainable rules. On the other hand, this approach could be further combined with other information like the type of change (vegetation - long term, dynamic - short term, etc.) and semantic information from the reference 3D map (e.g., vegetation, building, etc.). Also, since there are also HD maps more and more widely available in the industry, a complex method could include additional road structure information from HD maps (e.g., walk areas, driveable areas, etc.) to further improve the accuracy of the object detection and to provide an additional level of verification.

6.2.3 Temporal upscaling of spatially densified sparse Lidar measurements

In Chapter 5, we proposed a new method for the spatial densification of sparse measurements of a non-rotating circular scanning Lidar, by increasing the sensor's field-of-view coverage from around 40% to almost 100%. As the main limitation of the proposed technique, to have a reasonable spatial resolution of the raw Lidar depth data, we integrated our measurements within a 200 millisecond time window that results in a temporal resolution of 5 fps. According to our experiments, collecting the Lidar measurements with smaller time windows resulted in too sparse point clouds with a lack of spatial features. On the other hand, there are recent methods that focus on the temporal upscaling of Lidar point cloud measurements, using camera-based depth estimation techniques [101] or joint Lidar-camera approaches [102].

In this context, it would be a challenging task to apply temporal upscaling to the generated dense depth data stream, without the usage of reference camera images to further increase its temporal resolution, using interpolation or frame prediction techniques adopted from RGB video analysis methods.

The End.

Appendix A

Summary of the Thesis

This dissertation presents novel methods for real-time environment analysis using low-resolution Lidar point cloud measurements. I studied two different research problems: In the first topic, I focused on the fusion of onboard RMB and offline MLS data modalities for advanced urban scene analysis. I presented (i) a novel object-based point cloud matching algorithm for Lidar-based fast self-localization, and an extension of this algorithm by integrating its results to a Kalman-filter for accurate pose tracking, (ii) a novel change detection algorithm for low-level scene segmentation, and (iii) a high-level application of the above methods for improved dynamic object detection. In the second topic, I focused on the densification of sparse NRCS Lidar measurements without any prior information and I presented a novel depth completion method for enhancing the quality of sparse and noisy depth measurement data in real-time. All proposed methods have been evaluated in real-life urban traffic scenarios and experimentally compared against the state of the art, showing significant advantages.

A.1 New scientific results

1. Thesis: I have proposed a new method for change detection, which comprises 3D point cloud registration and point-level change segmentation through fusing Lidar point clouds with significantly different density characteristics. I have constructed a new urban dataset by a state-of-the-art rotating multi-beam (RMB) Lidar scanner (with a point density of around 50-500 points/ m^2) and an

up-to-date Mobile Laser Scanning (MLS) system (more than 5000 points/m^2). Using this new dataset, I have quantitatively demonstrated the advantage of the proposed algorithm against various state-of-the-art reference techniques.

Published in [1] [3] [4] [5]

This thesis deals with three consecutive subtasks: First, a cross-source point cloud registration is performed to precisely align the sparse RMB and dense MLS data, and a utilization of this algorithm is introduced for tracking the pose of the capturing vehicle in real time, even in partially occluded and dynamic environment. Second, a cross-source change detection algorithm is proposed for finding every point that changed since the map creation. Third, a new method is introduced by utilizing the output of the change detection technique for improved dynamic object detection.

1.1. I have proposed a novel approach for the registration of sparse RMB Lidar and dense MLS point clouds with a relatively poor initial alignment, which consists of a coarse pre-alignment step through detecting and matching landmark object candidates using geometry-based feature points from the whole scene, and a point-level refinement step that calculates the accurate transformation matrix based on the matched objects' local point cloud segments for reducing the computational need. I have demonstrated the advantage of the proposed algorithm against various state-of-the-art reference techniques in urban scenes, by comparing translation and rotation errors calculated by the decomposition of each transformation matrix and the manually labelled ground truth (GT), and by measuring point distances between the registered point clouds. I have shown an efficient utilization of the introduced algorithm for pose tracking, by estimating the planar pose of the vehicle from the object-matching result and fusing it in a constant velocity model-based Kalman filter.

Several point cloud registration algorithms exist in the literature that perform correspondences between features of handcrafted [31, 33, 34] or learning-based [35, 36] keypoints, segments [42, 43] or points [48–50]. In the addressed cross-source application, the main challenge is that the RMB Lidar frames are too sparse for extracting meaningful 3D keypoints (which work between dense MLS point clouds), while the MLS point clouds are in several regions 100-1000 times denser than the corresponding RMB measurement segments which misleads the general segment level matching processes. Following a different

approach, as the first contribution, instead of aligning the original point clouds, I have separated and matched landmark objects in the RMB Lidar frames and from the MLS map. Here, as a remaining challenge, many falsely detected object candidates can present (e.g., traffic participants, partially occluded objects), which may result in a possibly large ratio of outlier matches. To handle their effect, I have applied the voting schema of the generalized Hough transform. As the second contribution, I have used the coarse alignment step to initialize the point-level Iterative Closest Point algorithm [25], which I have executed only for point cloud segments corresponding to the previously aligned object pairs. In comparison to six different point cloud registration methods [28, 36, 40, 41, 47, 48], the median value of point-level distances is decreased by 1–2 orders of magnitude by the proposed approach.

To overcome the problem of heavily occluded scenarios without a sufficient number of matchable object pairs, I have extracted the planar (3DoF) pose (planar position and yaw orientation) of the capturing vehicle from the result of the object matching process and integrated the estimated pose parameters by a constant velocity (CV) model-based Kalman filter. Starting from a poor GPS-based positioning with 5-10 meters error, the proposed pose tracking approach is able to reduce the location error of the vehicle by one order of magnitude and to keep the yaw angle error around 1° during its whole trajectory without considerable drift, while running in real time (20-25 Hz).

1.2. I have proposed a new Markov Random Field-based approach (RangeMRF) for multi-class change extraction and classification (dynamic, seasonal, or no change) between registered RMB and MLS point clouds using 2D range image representations. I have demonstrated the advantage of the proposed algorithm against various state-of-the-art reference methods by qualitative and quantitative evaluations.

Three-dimensional change detection is a highly discussed topic in the literature [66], however, existing point [68, 69], segment [70] or voxel [72] based methods cannot handle well when the characteristics of the two comparable point sets are significantly different. In the addressed scenario, they show notable trade-off between false positive (e.g., noise, vegetation changes) and false negative hits (i.e., loss of details). As the first advantage, the proposed RangeMRF method detects changes between 2D range images derived from the point clouds. Using a compact range image representation, the proposed

method is notably quick, meanwhile, it can robustly handle the significantly different density characteristics of the two point sets by containing only relevant parts of the dense MLS data. Second, I have applied a Markov Random Field model, which is highly robust though noisy measurement data. Third, I have distinguished three classes in the segmentation model: seasonal changes in vegetation regions, foreground changes caused by moving objects or changed/re-located static street furniture elements, and unchanged background regions. By handling the vegetation areas with a specific sensitivity, the proposed method can eliminate several false hits in vegetation areas, while it is able to sharply recognize even small foreground changes (i.e., pedestrians standing near stations or facades) between the input point clouds. In comparison to four reference techniques [66, 69, 70, 72], the proposed method outperforms them either in F1-scores (by around 10-25%) or in computational complexity, running 10–1000 times faster.

1.3. I have proposed a new method to utilize the introduced change detection approach for improving the performance of Lidar-only dynamic object detection algorithms. I have demonstrated in high-traffic road sections that the proposed approach can efficiently balance the precision and recall values with significant overall improvement for both vehicles and pedestrians, using a state-of-the-art object detection method.

Real-time dynamic object detection in 3D sparse point clouds is a hot topic in autonomous driving with several geometric [52] and deep learning [51,74–77] based algorithms in the literature. However, there are a number of limitations of these approaches: False positive hits may appear in point cloud regions containing static scene objects with similar appearance and context parameters to the focused dynamic scene objects, while the point cloud blobs of several dynamic objects can be heavily merged or occluded by static street furniture elements, yielding many unrecognized traffic participants. I have proposed a new approach that utilizes dense MLS maps in order to decrease in parallel both the false negative and false positive hits of object detection algorithms. The proposed approach includes a map-based object validation, the introduced cross-source change extraction, and an object-level change analysis step between registered RMB and MLS map data. As a basis of comparison, I have chosen the *PointPillars* [51] state-of-the-art object detection method, with which the proposed method achieved an improvement of 5.96% in preci-

sion, 9.21% in recall and 7.93% in F1-score metrics on our own dataset.

2. Thesis: I have proposed a novel depth completion method from sparse consecutive measurements of a non-repetitive circular scanning (NRCS) Lidar using a deep learning model (ST-DepthNet). I have constructed a new urban dataset that comprises various simulated and real-world NRCS Lidar data samples. Using this new dataset, I have qualitatively and quantitatively demonstrated the superiority of the proposed method against a densified depth map obtained from the raw sensor stream, and against two independent state-of-the-art Lidar-only depth completion algorithms.

Published in [2], [7]

This thesis deals with efficient data-driven completion of NRCS Lidar data. Due to their non-repetitive scanning technology, NRCS Lidars are able to map different areas of their field of view (FoV) from a given scanning position in consecutive times [22]. The main challenge of analysing their point cloud streams is to efficiently balance between the spatial and the temporal resolution of the recorded range data using a suitable integration window. Allowing a larger integration time (e.g., 1 s) yields high spatial measurement resolution with various artifacts, such as blurred shapes of the observed vehicles, pedestrians or buildings, which phenomena complicate dynamic event analysis, while a narrow time window (e.g., 200 ms) yields spatially more precise but notably sparse measurements with a significant loss of spatial details. To overcome this spatio-temporal trade-off of the NRCS Lidar-based perception, I have proposed a novel deep learning-based approach for the densification of sparse NRCS Lidar depth data streams while keeping their temporal resolution and spatial accuracy high.

2.1. I have proposed a new training framework for the densification of 3D data streams provided by NRCS Lidars, by mapping their consecutive point cloud measurements to sparse 2D depth images, each collected within 200 ms to enable 40% field-of-view coverage. I have constructed a new synthetic dataset which contains depth images acquired by simulating the behaviour of a NRCS Lidar, and high-quality dense depth images for each sparse sample exploiting the complete spatial information of the virtual world. I have extended the dataset with sparse real samples using the same depth image representation and I have made them both publicly available. The proposed framework enables

to train and to test depth completion algorithms on synthetic scenarios, and to validate their reliability in real-world data as well.

For the 2D representation, I have converted the captured sparse point clouds of the Livox AVIA NRCS Lidar sensor from the Cartesian (x, y, z) to the spherical (distance, azimuth, elevation) polar coordinate system, then the horizontal and vertical FoVs were quantized onto a 400×400 pixel lattice using an integration time window of 200 ms for collecting the consecutive time frames. By this approach, around 60% of the range image pixels receive undefined range values, however, they are not notably effected by blurring and this representation permits the use of two-dimensional convolutional neural networks to fill in the missing structural information in the image domain. As higher integration time induces blurred silhouettes due to the independent movements of dynamic objects of the scene including the ego robot or vehicle, it is challenging to provide dense, spatially precise GT depth information for real data. Therefore, besides the real measurements, I have constructed a synthetic range image dataset from a realistic virtual world using the CARLA simulator [26], where the behaviour of the Livox AVIA NRCS Lidar sensor was implemented. The virtual world allows to extract dense, spatially precise depth information, used as GT for the Lidar's sparse depth sample data.

2.2. I have proposed a new depth completion deep neural network called ST-DepthNet, which extends the classical U-Net architecture with a spatio-temporal downscaling branch for utilizing five consecutive sparse measurements captured by NRCS Lidars and produces spatially precise high-density depth data in real time. I have demonstrated the advantage of the proposed algorithm against the state of the art in both synthetic and real-world scenarios.

First, I have exploited that using the applied Livox AVIA sensor, a time interval of 1 s contains enough dense range information from the scene with almost full FoV coverage. Thus, I have taken five consecutive sparse depth images – each one recorded in 200 ms – as the network's input to have enough information about the complete FoV. To accurately restore the single output image, I have adopted an image-to-image U-Net [81] architecture and I have extended the downscaling part of the U-Net network by utilizing Conv2DLSTM layers [82] to exploit temporal connections between the features derived from the input image sequence. Second, the upscaling branch of the proposed network remained purely two-dimensional and skip connections at each level were

performed by recurrent pooling utilizing the last output of a Conv2DLSTM layer which represents features of the last 200 ms measurement. Third, I have directly connected the last input image to the output to exploit that the last and most up-to-date input image contains spatially precise points, and therefore, the network only has to learn the missing regions of the range image. I have trained the model on the introduced synthetic dataset and I have quantitatively shown that the proposed approach outperforms two state-of-the-art reference methods [88,92] on synthetic data, reducing their root-mean-squared error (RMSE) by more than 1 meter in the range domain, and achieving around a half meter less error in the 3D domain by the normalized Chamfer distance and median distance. For real NRCS measurement data, a survey from 20 computer vision-related experts demonstrated that the proposed method performs significantly better than the reference techniques.

A.2 Application of the results

All the developed algorithms can be used in advanced perception platforms of mobile robots and intelligent vehicles equipped with RMB or NRCS Lidar sensors. The first thesis can be applied in urban environment where detailed 3D point cloud maps are available about the city for assisting vehicles that are equipped with a RMB Lidar and a GPS receiver. The proposed methods can contribute to map-based real-time scene understanding like accurate self-localization and pose tracking, change-based scene segmentation and improved dynamic object detection. The second thesis can be applied for robot or vehicle platforms that are equipped with a NRCS Lidar sensor to produce accurate and dense depth maps from the environment while keeping high temporal resolution, which can be a crucial middle step for more advanced scene understanding or mapping. As part of my Cooperative Doctoral Program, many of the proposed algorithms directly contributed to R&D projects conducted with the participation of the Institute for Computer Science and Control (SZTAKI) and the Pázmány Péter Catholic University (PPCU), and we also submitted two patent applications related to the methods.

Appendix B

List of Abbreviations

2D	Two-dimensional
3D	Three-dimensional
ADAS	Advanced Driving Assistant System
AV	Autonomous Vehicle
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CV	Constant Velocity
DoF	Degree of Freedom
FoV	Field of View
GIS	Geo-Information System
GPS	Global Positioning System
GPU	Graphical Processing Unit
GT	Ground Truth
HD	High Definition
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
Lidar	Light Detection and Ranging
LSTM	Long-Short Term Memory

MAE	Mean Absolute Error
MHD	Modified Hausdorff Distance
MLS	Mobile Laser Scanning
MPD	Median Point Distance
MRF	Markov Random Field
NCD	Normalized Chamfer Distance
NIR	Near Infrared
NMD	Normalized Median Distance
NN	Nearest Neighbor
NRCS	Non-repetitive Circular Scanning
OD	Object Detection
POM	Position-only-Measured
Radar	Radio detection and ranging
RANSAC	Random Sample Consensus
RMB	Rotating Multi-beam
RMSE	Root Mean Squared Error
SLAM	Simultaneous Localization and Mapping
ST	Spatio-temporal
SVM	Support Vector Machine
SWIR	Short-wave Infrared
TLS	Terrestrial Laser Scanning
ToF	Time-of-Flight
TIR	Thermal Infrared

Appendix C

List of Figures

Appendix D

List of Tables

Journal publications of the Thesis

[1] **Ö. Zováthi**, B. Nagy, and C. Benedek, "Point cloud registration and change detection in urban environment using an onboard lidar sensor and MLS reference data," *International Journal of Applied Earth Observation and Geoinformation*, vol. 110, p. 102767, 2022.

[2] **Ö. Zováthi**, B. Pálffy, Z. Jankó, and C. Benedek, "ST-DepthNet: A spatio-temporal deep network for depth completion using a single non-repetitive circular scanning lidar," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3270–3277, 2023.

Conference publications of the Thesis

[3] **Ö. Zováthi**, B. Pálffy, and C. Benedek, "Real-time vehicle localization and pose tracking in high-resolution 3D maps," in *30th European Signal Processing Conference (EUSIPCO)*, 2022, pp. 1786–1790.

[4] **Ö. Zováthi**, B. Nagy, and C. Benedek, "Exploitation of dense MLS city maps for 3D object detection," in *International Conference on Image Analysis and Recognition (ICIAR)*, ser. Lecture Notes in Computer Science. Springer, 2020, vol. 12131, pp. 393–403.

[5] **Ö. Zováthi**, L. Kovács, B. Nagy, and C. Benedek, "Multi-object detection in urban scenes utilizing 3D background maps and tracking," in *International Conference on Control, Artificial Intelligence, Robotics and Optimization (ICCAIRO)*, 2019, pp. 231–236.

Patents related to the Thesis

[6] B. Nagy, L. Kovács, C. Benedek, T. Szirányi, **Ö. Zováthi**, and L. Tizedes, "Training method for training a change detection system, training set generating method therefor, and change detection system," WO Patent application, WO/2023/007198, International Filing Date: 08.07.2022, Priority Data: P2100280, 27.07.2021, HU.

[7] **Ö. Zováthi**, Z. Rózsa, B. Pálffy, Z. Jankó, C. Benedek, T. Szirányi, L. Kovács, and M. Kégl, "Methods for spatial and temporal densification of Lidar measurements," Patent application, Priority Data: P2300075, 01.03.2023, HU.

Bibliography

[8] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *IEEE Conference on Computer Vision and Pattern Recognition*, Providence, RI, USA, 05 2012, pp. 3354–3361.

[9] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "NuScenes: a multimodal dataset for autonomous driving," *arXiv preprint arXiv:1903.11027*, 2019.

[10] T. Matuszka, I. Barton, Ádám Butykai, P. Hajas, D. Kiss, D. Kovács, S. Kunsági-Máté, P. Lengyel, G. Németh, L. Pető, D. Ribli, D. Szeghy, S. Vajna, and B. Varga, "aiMotive dataset: A multimodal dataset for robust autonomous driving with long-range perception," *arXiv preprint arXiv:2211.09445*, 2023.

[11] C. Benedek, A. Majdik, B. Nagy, Z. Rózsa, and T. Szirányi, "Positioning and perception in LIDAR point clouds," *Digital Signal Processing*, vol. 119, p. 103193, 2021.

[12] S. Gu, Y. Zhang, J. Tang, J. Yang, J. M. Alvarez, and H. Kong, "Integrating dense lidar-camera road detection maps by a multi-modal crf model," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 12, pp. 11 635–11 645, 2019.

[13] P. Wang, "Research on comparison of lidar and camera in autonomous driving," *Journal of Physics: Conference Series*, vol. 2093, p. 012032, 11 2021.

[14] B. Bayat, N. Crasta, A. Crespi, A. M. Pascoal, and A. Ijspeert, "Environmental monitoring using autonomous vehicles: a survey of recent

searching techniques," *Current Opinion in Biotechnology*, vol. 45, pp. 76 – 84, 2017.

[15] X. Mi, B. Yang, Z. Dong, C. Chen, and J. Gu, "Automated 3D road boundary extraction and vectorization using MLS point clouds," *IEEE Transactions on Intelligent Transportation Systems*, vol. PP, pp. 1–11, 01 2021.

[16] W. Ma, I. Tartavull, I. A. Barsan, S. Wang, M. Bai, G. Mattyus, N. Homayounfar, S. K. Lakshmikanth, A. Pokrovsky, and R. Urtasun, "Exploiting sparse semantic HD maps for self-driving vehicle localization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Macau, China, 11 2019, pp. 5304–5311.

[17] H. Zheng, R. Wang, and S. Xu, "Recognizing street lighting poles from mobile lidar data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 1, pp. 407–420, Jan 2017.

[18] H. Guan, J. Z. Li, Y. Yu, M. A. Chapman, and C. Wang, "Automated road information extraction from mobile laser scanning data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, pp. 194–205, 02 2015.

[19] B. Nagy and C. Benedek, "3D CNN-based semantic labeling approach for mobile laser scanning data," *IEEE Sensors Journal*, vol. 19, no. 21, pp. 10 034–10 045, Nov 2019.

[20] P. F. McManamon, *LiDAR Technologies and Systems*. SPIE, Jul 2019.

[21] C. Glennie and P. Hartzell, "Accuracy assessment and calibration of low-cost autonomous lidar sensors," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences (ISPRS)*, vol. XLIII-B1-2020, pp. 371–376, 08 2020.

[22] L. Kovács, M. Kégl, and C. Benedek, "Real-time foreground segmentation for surveillance applications in NRCS lidar sequences," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences (ISPRS)*, vol. XLIII-B1-2022, pp. 45–51, 05 2022.

[23] A. Patil, S. Malla, H. Gang, and Y.-T. Chen, "The H3D dataset for full-surround 3D multi-object detection and tracking in crowded urban

scenes," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[24] R. B. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, 2011, pp. 1–4.

[25] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *International Journal of Computer Vision*, vol. 13, no. 2, pp. 119–152, 1994.

[26] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

[27] Y. Kwon, M. Sung, and S. Yoon, "Implicit lidar network: Lidar super-resolution via interpolation weight prediction," in *IEEE International Conference on Robotics and Automation*, 2022, pp. 8424–8430.

[28] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-D point sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, no. 5, pp. 698–700, 1987.

[29] L. Cheng, S. Chen, X. Liu, H. Xu, Y. Wu, M. Li, and Y. Chen, "Registration of laser scanning point clouds: A review," *MDPI Sensors*, vol. 18, p. 1641, 05 2018.

[30] X. Ge, H. Hu, and B. Wu, "Image-guided registration of unordered terrestrial laser scanning point clouds for urban scenes," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 11, pp. 9264–9276, 2019.

[31] R. Hänsch, T. Weber, and O. Hellwich, "Comparison of 3D interest point detectors and descriptors for point cloud fusion," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. II-3, pp. 57–64, 09 2014.

[32] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3D registration," in *IEEE International Conference on Robotics and Automation (ICRA)*, Kobe, Japan, May 2009, pp. 3212–3217.

[33] I. Sipiran and B. Bustos, "Harris 3D: A robust extension of the Harris operator for interest point detection on 3D meshes," *The Visual Computer*, vol. 27, pp. 963–976, 11 2011.

[34] Y. Zhong, "Intrinsic shape signatures: A shape descriptor for 3D object recognition," in *IEEE International Conference on Computer Vision (ICCV) Workshops*, Kyoto, Japan, 11 2009, pp. 689–696.

[35] J. Li and G. H. Lee, "USIP: Unsupervised stable interest point detection from 3D point clouds," in *IEEE/CVF IEEE International Conference on Computer Vision (ICCV)*, Seoul, Korea, 2019, pp. 361–370.

[36] F. Lu, G. Chen, Y. Liu, Z. Qu, and A. Knoll, "RSKDD-Net: Random sample-based keypoint detector and descriptor," in *Advances in Neural Information Processing Systems*, vol. 33, virtual Conf., 2020, pp. 21 297–21 308.

[37] P. W. Theiler, J. D. Wegner, and K. Schindler, "Keypoint-based 4-points congruent sets – automated marker-less registration of laser scans," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 96, pp. 149 – 163, 2014.

[38] Y. Li and E. B. Olson, "Extracting general-purpose features from LIDAR data," in *IEEE International Conference on Robotics and Automation (ICRA)*, Anchorage, AK, USA, 2010, pp. 1388–1393.

[39] Z. Dong, B. Yang, Y. Liu, F. Liang, B. Li, and Y. Zang, "A novel binary shape context for 3D local surface description," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 130, pp. 431 – 452, 2017.

[40] S. Choi, Q. Zhou, and V. Koltun, "Robust reconstruction of indoor scenes," in *IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, USA, 2015, pp. 5556–5565.

[41] H. Yang, J. Shi, and L. Carlone, "Teaser: Fast and certifiable point cloud registration," *IEEE Transactions on Robotics*, vol. 37, no. 2, pp. 314–333, 2021.

[42] B. Douillard, A. Quadros, P. Morton, J. P. Underwood, M. D. Deuge, S. Hugosson, M. Hallström, and T. Bailey, "Scan segments matching for

pairwise 3D alignment," in *IEEE International Conference on Robotics and Automation (ICRA)*, St. Paul, MN, USA, May 2012, pp. 3033–3040.

[43] R. Dubé, A. Cramariuc, D. Dugas, J. Nieto, R. Siegwart, and C. Cadena, "SegMap: 3D segment mapping using data-driven descriptors," in *Proceedings of Robotics: Science and Systems*, 06 2018.

[44] X. Huang, G. Mei, J. Zhang, and R. Abbas, "A comprehensive survey on point cloud registration," *arXiv preprint arXiv:2103.02690*, 2021.

[45] X. Huang, J. Zhang, L. Fan, Q. Wu, and C. Yuan, "A systematic approach for cross-source point cloud registration by preserving macro and micro structures," *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3261–3276, 2017.

[46] X. Huang, L. Fan, Q. Wu, J. Zhang, and C. Yuan, "Fast registration for cross-source point clouds by using weak regional affinity and pixel-wise refinement," in *2019 IEEE International Conference on Multimedia and Expo (ICME)*. Los Alamitos, CA, USA: IEEE Computer Society, jul 2019, pp. 1552–1557.

[47] X. Huang, G. Mei, and J. Zhang, "Feature-metric registration: A fast semi-supervised approach for robust point cloud registration without correspondences," in *IEEE Computer Vision and Pattern Recognition Conference*, June 2020.

[48] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *International Journal of Computer Vision*, vol. 13, no. 2, p. 119–152, Oct. 1994.

[49] A. Gressin, C. Mallet, and N. David, "Improving 3D LIDAR Point Cloud Registration Using Optimal Neighborhood Knowledge," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. I-3, pp. 111–116, 08 2012.

[50] A. Gressin, B. Cannelle, C. Mallet, and J.-P. Papelard, "Trajectory-Based Registration of 3D LIDAR Point Clouds Acquired with a Mobile Mapping System," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. I-3, pp. 117–122, 08 2012.

[51] A. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "PointPillars: Fast Encoders for Object Detection from Point Clouds," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, 06 2019, pp. 12 689–12 697.

[52] A. Börcs, B. Nagy, and C. Benedek, "Instant object detection in lidar point clouds," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 7, pp. 992–996, July 2017.

[53] N. K. Ratha, K. Karu, S. Chen, and A. K. Jain, "A real-time matching system for large fingerprint databases," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 799–813, Aug 1996.

[54] B. Nagy and C. Benedek, "Real-time point cloud alignment for vehicle localization in a high resolution 3D map," in *European Conference on Computer Vision (ECCV) Workshops*, 2019, pp. 226–239.

[55] J. Park, P. Kim, Y. Cho, and J. Kang, "Framework for automated registration of UAV and UGV point clouds using local features in images," *Automation in Construction*, vol. 98, pp. 175–182, 02 2019.

[56] V. Lehtola, H. Kaartinen, A. Nuchter, R. Kaijaluoto, A. Kukko, P. Litkey, E. Honkavaara, T. Rosnell, M. Vaaja, J.-P. Virtanen, M. Kurkela, A. Issaoui, L. Zhu, A. Jaakkola, and J. Hyyppä, "Comparison of the selected state-of-the-art 3D indoor scanning and point cloud generation methods," *MDPI Remote Sensing*, vol. 9, p. 796, 08 2017.

[57] J. Zhang, X. Chen, Z. Cai, L. Pan, H. Zhao, S. Yi, C. Yeo, B. Dai, and C. C. Loy, "Unsupervised 3d shape completion through gan inversion," in *Computer Vision and Pattern Recognition (CVPR)*, 06 2021, pp. 1768–1777.

[58] A. Saint, A. Kacem, K. Cherenkova, and D. Aouada, "3dbooster: 3d body shape and texture recovery," in *European Conference on Computer Vision (ECCV) 2020 Workshops*, A. Bartoli and A. Fusiello, Eds., 2020, pp. 726–740.

[59] D. Qiu, S. May, and A. Nüchter, "GPU-accelerated nearest neighbor search for 3D registration," in *7th International Conference on Computer Vision Systems*, Liege, Belgium, 10 2009, pp. 194–203.

[60] D. Rozenberszki and A. Majdik, "Lol: Lidar-only odometry and localization in 3D point cloud maps," *IEEE International Conference on Robotics and Automation*, pp. 4379–4385, 2020.

[61] J. Zhang and S. Singh, "LOAM: lidar odometry and mapping in real-time," in *Robotics: Science and Systems Conference (RSS) X*, 2014.

[62] S. Scherer, J. Rehder, S. Achar, H. Cover, A. Chambers, S. Nuske, and S. Singh, "River mapping from a flying robot: state estimation, river detection, and obstacle mapping," *Autonomous Robots*, vol. 33, no. 1, pp. 189–214, Aug 2012.

[63] W. Burgard, O. Brock, and C. Stachniss, *Map-Based Precision Vehicle Localization in Urban Environments*, 2008, pp. 121–128.

[64] C. Benedek, *Multi-Level Bayesian Models for Environment Perception.* Springer International Publishing, 2022.

[65] T. Ku, S. Galanakis, B. Boom, R. C. Veltkamp, D. Bangera, S. Gangisetty, N. Stagakis, G. Arvanitis, and K. Moustakas, "SHREC 2021: 3D point cloud change detection for street scenes," *Computers and Graphics*, vol. 99, pp. 192–200, 2021.

[66] W. Xiao, B. Vallet, M. Brédif, and N. Paparoditis, "Street environment change detection from mobile laser scanning point clouds," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 107, pp. 38 – 49, 2015.

[67] H. Aljumaily, D. F. Laefer, and D. Cuadra, "Urban point cloud mining based on density clustering and mapreduce," *Journal of Computing in Civil Engineering*, vol. 31, no. 5, p. 04017021, 2017.

[68] D. Liu, D. Li, M. Wang, and Z. Wang, "3D change detection using adaptive thresholds based on local point cloud density," *ISPRS International Journal of Geo-Information*, vol. 10, no. 3, 2021.

[69] D. Girardeau-Montaut, M. Roux, R. Marc, and G. Thibault, "Change detection on point cloud data acquired with a ground laser scanner," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences (ISPRS)*, vol. 36, 01 2005.

[70] A. Schlichting and C. Brenner, "Vehicle localization by lidar point correlation improved by change detection," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences (ISPRS)*, vol. XLI-B1, pp. 703–710, 06 2016.

[71] M. Voelsen, J. Schachtschneider, and C. Brenner, "Classification and change detection in mobile mapping lidar point clouds," *Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, vol. 89, no. 3, pp. 195–207, Jun 2021.

[72] K. Liu, J. Boehm, and C. Alis, "Change detection of mobile lidar data using cloud computing," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences (ISPRS)*, vol. XLI-B3, pp. 309–313, 06 2016.

[73] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1124–1137, Sep. 2004.

[74] K. Shin, Y. Kwon, and M. Tomizuka, "RoarNet: A robust 3D object detection based on region approximation refinement," *IEEE Intelligent Vehicles Symposium*, pp. 2510–2515, 2018.

[75] M. Simon, S. Milz, K. Amende, and H.-M. Groß, "Complex-YOLO: Real-time 3D object detection on point clouds," *arXiv preprint arXiv:1803.06199*, 2018.

[76] Y. Yan, Y. Mao, and B. Li, "SECOND: Sparsely embedded convolutional detection," *MDPI Sensors*, vol. 18, p. 3337, 10 2018.

[77] B. Yang, W. Luo, and R. Urtasun, "PIXOR: Real-time 3D object detection from point clouds," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7652–7660.

[78] B. Yang, M. Liang, and R. Urtasun, "HDNET: Exploiting HD maps for 3D object detection," in *Proceedings of the 2nd Conference on Robot Learning*, vol. 87, 29–31 Oct 2018, pp. 146–155.

[79] J. Fang, D. Zhou, X. Song, and L. Zhang, "Mapfusion: A general framework for 3d object detection with hdmaps," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3406–3413, 2021.

[80] K. Kidono, T. Miyasaka, A. Watanabe, T. Naito, and J. Miura, "Pedestrian recognition using high-definition LIDAR," *Journal of the Robotics Society of Japan*, vol. 29, pp. 405–410, 2011.

[81] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, 2015, pp. 234–241.

[82] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[83] T. Shan, J. Wang, F. Chen, P. Szenher, and B. Englot, "Simulation-based lidar super-resolution for ground vehicles," *Robotics and Autonomous Systems*, vol. 134, p. 103647, 2020.

[84] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997.

[85] J. Uhrig, N. Schneider, L. Schneider, U. Franke, T. Brox, and A. Geiger, "Sparsity invariant CNNs," in *International Conference on 3D Vision (3DV)*, 2017.

[86] S. Zhao, M. Gong, H. Fu, and D. Tao, "Adaptive context-aware multimodal network for depth completion," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 30, pp. 5264–5276, 2021.

[87] D. Nazir, A. Pagani, M. Liwicki, D. Stricker, and M. Z. Afzal, "SemAttNet: Towards attention-based semantic aware guided depth completion," *IEEE Access*, pp. 1–1, 2022.

[88] M. F. F. Khan, N. D. Troncoso Aldas, A. Kumar, S. Advani, and V. Narayanan, "Sparse to dense depth completion using a generative adversarial network with intelligent sampling strategies," in *Proceedings of*

the 29th ACM International Conference on Multimedia, New York, NY, USA, 2021, p. 5528–5536.

[89] R. Li, D. Xue, Y. Zhu, H. Wu, J. Sun, and Y. Zhang, "Self-supervised monocular depth estimation with frequency-based recurrent refinement," *IEEE Transactions on Multimedia*, pp. 1–12, 2022.

[90] A. Savkin, Y. Wang, S. Wirkert, N. Navab, and F. Tombari, "Lidar upsampling with sliced wasserstein distance," *IEEE Robotics and Automation Letters*, pp. 1–8, 2022.

[91] C. Godard, O. Aodha, M. Firman, and G. Brostow, "Digging into self-supervised monocular depth estimation," in *IEEE International Conference on Computer Vision (ICCV)*, 11 2019.

[92] J. Ku, A. Harakeh, and S. L. Waslander, "In defense of classical image processing: Fast depth completion on the CPU," *Canadian Conference on Computer and Robot Vision (CRV)*, pp. 16–22, 2018.

[93] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[94] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015.

[95] J. Zhao, X. He, J. Li, T. Feng, C. Ye, and L. Xiong, "Automatic Vector-Based Road Structure Mapping Using Multibeam LiDAR," *MDPI Remote Sensing*, vol. 11, p. 1726, 2019.

[96] C. Ye, H. Zhao, L. Ma, H. Jiang, H. Li, R. Wang, M. A. Chapman, J. M. Junior, and J. Li, "Robust lane extraction from MLS point clouds towards HD maps especially in curve road," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 2, pp. 1505–1518, 2022.

[97] C. Qi, L. Yi, H. Su, and L. Guibas, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," in *Conference on*

Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 2017, pp. 5105–5114.

[98] H. Su, V. Jampani, D. Sun, S. Maji, E. Kalogerakis, M.-H. Yang, and J. Kautz, "SPLATNet: Sparse lattice networks for point cloud processing," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 2530–2539.

[99] E. Barçon and A. Picard, "Automatic detection and vectorization of linear and point objects in 3d point cloud and panoramic images from mobile mapping system," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLIII-B2-2021, pp. 305–312, 06 2021.

[100] L. Ma, Y. Li, J. Li, Z. Zhong, and M. A. Chapman, "Generation of horizontally curved driving lines in hd maps using mobile laser scanning point clouds," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 12, no. 5, pp. 1572–1586, 2019.

[101] S. Shao, Z. Pei, W. Chen, X. Wu, and Z. Li, "Nddepth: Normal-distance assisted monocular depth estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2023, pp. 7931–7940.

[102] Z. Rozsa and T. Sziranyi, "Optical flow and expansion based deep temporal up-sampling of lidar point clouds," *Remote Sensing*, vol. 15, no. 10, 2023.

Image sources and web references

[103] Velodyne Lidar, "Lidar ready to help decrease pedestrian roadway perils," URL: https://velodynelidar.com/blog/lidar-and-pedestrian-safety/.

[104] Velodyne Lidar, "Velodyne HDL64E Lidar sensor," URL: https://velodynelidar.com/.

[105] Ouster Lidar, "OS1 – Mid-range digital Lidar sensor," URL: https://ouster.com/products/scanning-lidar/os1-sensor/.

[106] Livox Lidar, "AVIA Lidar sensor," URL: https://www.livoxtech.com/avia.

[107] Riegl Measurement Systems, "Riegl VMX450," URL: www.riegl.com/uploads/tx_pxpriegldownloads/DataSheet_VMX-450_2015-03-19.pdf.

[108] "C++ Programming Language," URL: https://cplusplus.com/.

[109] "OpenCV – Open Computer Vision Library," URL: https://opencv.org/.

[110] "Point Cloud Library (PCL)," URL: https://pointclouds.org/.

[111] "Python Programming Language," URL: https://www.python.org/.

[112] "PyTorch," URL: https://pytorch.org/.

[113] "Keras," URL: https://keras.io/.

[114] "Robot Operating System (ROS)," URL: https://www.ros.org/.

All links were accessed on 1st November 2023 and were correct at the time of publication.