Data frame aware optimized Octomap-based dynamic object detection and removal in Mobile Laser Scanning data

Liu, Zhenyu; van Oosterom, Peter; Balado, Jesús; Swart, Arjen; Beers, Bart

DOI
10.1016/j.aej.2023.05.014

Publication date
2023

Document Version
Final published version

Published in
Alexandria Engineering Journal

Citation (APA)
Liu, Z., van Oosterom, P., Balado, J., Swart, A., & Beers, B. (2023). Data frame aware optimized Octomap-based dynamic object detection and removal in Mobile Laser Scanning data. *Alexandria Engineering Journal*, *74*, 327-344. https://doi.org/10.1016/j.aej.2023.05.014

Alexandria Engineering Journal (2023) 74, 327–344

HOSTED BY

Alexandria University

Alexandria Engineering Journal

www.elsevier.com/locate/aej
www.sciencedirect.com

ORIGINAL ARTICLE

Data frame aware optimized Octomap-based dynamic object detection and removal in Mobile Laser Scanning data

Zhenyu Liu [a,b,*], Peter van Oosterom [a], Jesús Balado [a,c], Arjen Swart [d], Bart Beers [d]

[a] GIS Technology, Faculty of Architecture and the Built Environment, Delft University of Technology, 2628 BL Delft, the Netherlands
[b] Geodetic Institute and Chair for Computing in Civil Engineering & Geo Information Systems, RWTH Aachen University, Mies-van-der-Rohe-Str. 1, 52074 Aachen, Germany
[c] GeoTECH Group, CINTECX, Universidade de Vigo, 36310 Vigo, Spain
[d] CycloMedia Technology B.V., Waardenburg, the Netherlands

Received 21 December 2022; revised 10 April 2023; accepted 3 May 2023

KEYWORDS

Mobile Laser Scanning;
LiDAR Data;
Point Cloud;
Octomap;
Dynamic Object Detection;
Dynamic Object Removal

Abstract The Mobile Laser Scanning (MLS) data inevitably includes dynamic objects because there are always other vehicles (e.g., other cars, motorbikes, bikes, etc.) moving in the area near the MLS data collection vehicle on the road. These dynamic objects need to be removed in advance for many point cloud applications. This paper designs an efficient and memory-friendly data frame aware optimized Octomap-based dynamic object detection and removal method for MLS data. Firstly, the input MLS data is split into multiple data frames based on the timestamp. Each data frame is inserted into a separate Octomap with part of its neighbouring data frames. A statistics-based method is applied to each data frame to find the passable voxel cell space (free space) in Octomap and all points in the free space are extracted as free points. Second, the region of interest (ROI) related to the dynamic object is delineated to retain free points related to dynamic objects. Then the free-point rate and the multi-return rate are calculated to further remove noise and vegetation points from free points. Finally, the fixed radius search is used to extract dynamic objects from the filtered free points. The proposed method is tested in four case sites in Delft, the Netherlands. Results show that 84.98% of dynamic objects are detected and extracted correctly. The proposed method is 18.27% more efficient on average than the original Octomap method, can be further accelerated by parallel computing, and only needs 39.40% of the maximum memory consumption.

1. Introduction

According to the motion state of objects during scanning, objects in Mobile Laser Scanning (MLS) data can be classified as static objects (like roads, buildings, vegetation, and parked

* Corresponding author at: GIS Technology, Faculty of Architecture and the Built Environment, Delft University of Technology, 2628 BL Delft, the Netherlands.
E-mail address: Z.LIU-46@student.tudelft.nl (Z. Liu).
Peer review under responsibility of Faculty of Engineering, Alexandria University.

https://doi.org/10.1016/j.aej.2023.05.014

vehicles) and dynamic objects (like pedestrians, cyclists, and moving vehicles, see Fig. 1.(a)). Only static environments are critical and necessary for many MLS engineering applications especially in the intelligent transportation and civil engineering fields, such as civil infrastructure investigation, built-up area change detection, HD-map generation, navigation, and location services [5,7,36,52,62]. If these applications use MLS data that contains dynamic objects, their final performance is affected. For example, residual dynamic objects reduce the location accuracy of point-based HD-map [18,57].

However, it is impossible to avoid dynamic objects in the original MLS data. The ghost trail effect [43,48] makes the problem of dynamic objects to be further aggravated. The moving route and speed of the MLS sensor are restricted by traffic laws, road networks, and other factors [6]. Therefore, the MLS sensor and its nearby dynamic objects are highly likely to move with similar speeds and trajectories. These dynamic objects are continuously scanned by their surrounding MLS sensors, which makes them to be very seriously stretched in the collected MLS data. In addition, MLS sensors and their surrounding dynamic objects' speed and moving direction are constantly changing, making the shape of the captured dynamic object to be bent during the stretching process (Fig. 1.(b)). Above factors result in many detection methods designed for Airborne Laser Scanning (ALS) and Terrestrial Laser Scanning (TLS) data cannot be directly applied to MLS data. So, it is more challenging to accurately detect dynamic objects in MLS data.

This research aims to design a data frame aware optimized Octomap-based dynamic object detection and removal method, which can be accelerated with parallel computing. The proposed method first split the input MLS data into multiple data frames based on the timestamp. Each data frame and part of its neighbouring data frames are inserted into a separate Octomap to extract all free points (i.e., points in the free space of Octomap). Then sensor trajectory, sensor mounting height, and the local large vehicle height restriction are used to delimitate the Region of Interest (ROI) for removing irrelevant free points. Next, the noise and vegetation points are filtered from the remaining free points based on the free-point rate and the multi-return rate. Finally, dynamic objects are extracted and removed by using the fixed radius search with filtered free points.

The paper is organized as follows: First Section 2 reviews previous related research about dynamic object detection and change detection in point cloud data. Section 3 presents our proposed methods. Then Section 4 indicates experimental details, show final experimental results, and analyzes the performance of our proposed methods. Finally, Section 5 gives conclusions of this work.

2. Related work

The detection and removal of point cloud dynamic objects are of great significance in many fields, such as autonomous driving [38], 3D mapping [3], and environmental monitoring [41,53]. The current methods are divided into single-frame data (Section 2.1) and multi-frame data (Section 2.2) methods according to whether multiple scans (or continuous scans in MLS) are required. Section 2.3 summarizes the progress and main problems of Octomap-related methods. Finally, Section 2.4 enumerates the improvements of our proposed method over existing methods.

2.1. Single-frame data methods

The single-frame LiDAR data means the scan data obtained by the sensor in a very short period. ALS and TLS scan results are usually single-frame data. For MLS, the data obtained by a single rotation of the sensor (360°) is also considered single-frame data.

The simplest single-frame data method for change detection is the prior map (background subtraction) method [1,30,37,64]. However, the generation and updating of the prior map also need to remove dynamic objects. So prior map construction and dynamic object detection are chicken-and-egg problems due to their interwoven nature [29]. Another problem is that it is difficult to accurately represent non-rigid background objects such as tree crowns and grasslands in the prior map [59].

Another idea based on prior knowledge is to extract potential moving objects using feature or model matching [8,12,16]. However, a large number of features and models need to be defined to cover all types of dynamic objects [23,28,33], which

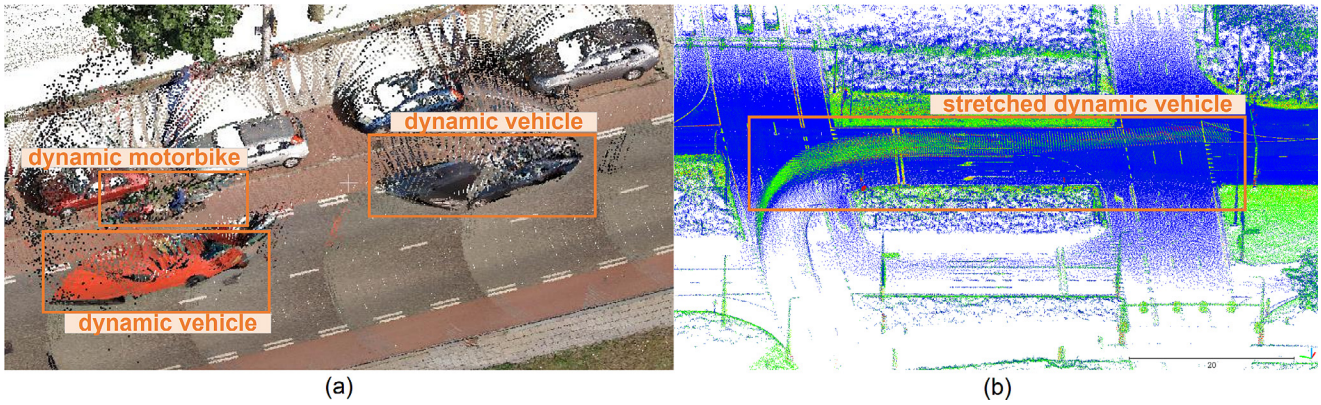

Fig. 1 Common dynamic objects in the road environment (a) and stretched dynamic objects in MLS data (b).

significantly increases the computational cost and memory consumption [56]. Besides, the ghost trail effect in MLS data makes it difficult to define accurate features and models for dynamic objects.

2.2. Multi-frame data methods

The multi-frame LiDAR data is a collection of multiple single-frame data, such as the continuously scanning MLS data. Spatiotemporal correlations among consecutive data frames provide much useful information for the detection of dynamic objects [27,35].

A common multi-frame method is object tracking [45,51]. Such methods require prior knowledge of the target object, so they usually have poor generalization ability and are not suitable for detecting unknown dynamic objects. To avoid the limitations of prior models, some model-free methods are also proposed. For example, Dewan et al. [15] first used RANSAC [19] to estimate the motion model and then segmented the point cloud directly with the motion cues. Zhang and Nakamura [67] obtained the dynamic objects using the inter-frame subtraction and then extracted the moving cues from their proposed Mean Axis Descriptor (MAD).

Some latest studies have proposed point-based dynamic detection. Therefore, dynamic tracing does not need to take the object as the basic unit but is further achieved at the point level [10,65]. But point-based methods are more susceptible to viewpoint occlusions and data sparsity than object-based methods.

Recently, learning-based methods have also been widely applied to this topic and show excellent performance [11,31,47,63]. But the good performance of such methods requires the support of high-quality training sets. For dynamic object detection, the relevant high-quality available outdoor data sets are still not enough in general. This problem can be partially solved by transfer learning from existing datasets, but it still requires a lot of human and computing resources [44,62]. For large open data sets such as KITTI [21], training samples, especially uncommon moving objects, are often imbalanced. This may affect the quality of the results of multi-type object recognition tasks [58].

2.3. Octomap-related methods

A common problem in previous methods is that the effect of noise [2,60] is ignored. Therefore, some studies use Octomap to detect dynamic objects [3,32,43,50,54]. Octomap reconstructs LiDAR rays of the point cloud in a voxel grid and extracts dynamic objects by counting hits and misses of LiDAR rays for each voxel space. Instead of simply marking points as static or dynamic, Octomap describes the probability of each voxel being occupied, which considers the effect of noise on ray tracing.

Although Octomap has improved the computational efficiency and memory consumption compared with previous methods, it still has weakness when dealing with the point cloud data with high density and large scan area. The insertion of large size outdoor MLS data into an Octomap usually results in a huge voxel grid, resulting in slow voxel cell iteration speed, large memory consumption [17,26]. This means that using high-resolution Octomap for outdoor MLS data to obtain accurate detection results requires a significant investment of computational and memory resources. Several studies optimized the performance of Octomap with parallel computation [25,39]. But these methods were only tested in small indoor environments and still cannot avoid generating very large voxel grids in open outdoor environments. Buerkle et al. [9] proposed a non-uniform grid structure to replace the original Octomap, but this requires well-performing algorithms to ensure that space occupancy probabilities are accurately transformed between grids of different resolutions.

2.4. Contributions of proposed method

The contributions of this study concerning the earlier work and the original Octomap method are summarized as follows:

(1) Introducing the data frame into Octomap method.
(2) Multiple smaller Octomaps are generated to replace one huge Octomap.
(3) Improvements in computational efficiency and memory consumption of Octomap.
(4) Local vehicle height restriction is used to reduce the upper boundary of ROI.
(5) Integration with parallel computing for better performance.

3. Methodology

The methodology of this research (Fig. 2) starts with extracting free points from Octomaps (Section 3.1). Then the ROI is delimitated to filter relevant free points (Section 3.2). Next, noise (Section 3.3) and vegetation points (Section 3.4) are detected and removed. Finally, dynamic objects are extracted from original MLS data using a fixed radius search (Section 3.5).

3.1. Free point extraction

Octomap is a state-of-the-art octree-based probabilistic occupancy voxel grid structure. Fig. 3.(a) illustrates the main idea of dividing the whole point clouds space into the passable space (free space) and non-passable space (occupied space) using Octomap. It first converts the input point clouds into a voxel grid and then traces the LiDAR rays in the grid. For each LiDAR ray $s_i p_i$, the voxel cell v_{p_i} containing the captured point p_i belongs to the occupied space. The voxel cell v_{s_i} containing the MLS sensor s_i and all voxel cells between v_{p_i} and v_{s_i} belong to the free space. The others are unscanned cells. However, sometimes there is a spatial conflict between different rays, making the space not so easily to be divided. Fig. 3.(b) shows a typical spatial conflict: Captured point p_2 is located in the middle of the ray $s_1 p_1$. So for ray $s_1 p_1$, voxel cell v_{p_2} is free space. While for ray $s_2 p_2$, v_{p_2} is occupied space. This conflict is caused by many potential factors, such as accuracy loss due to ray voxelization, measurement errors of LiDAR sensors, objects with sparse structure, non-rigid objects, or dynamic objects. Without knowing which factor is the real cause, v_{p_2} cannot be clearly classified as free or occupied space. For this issue, Octomap does not directly deter-

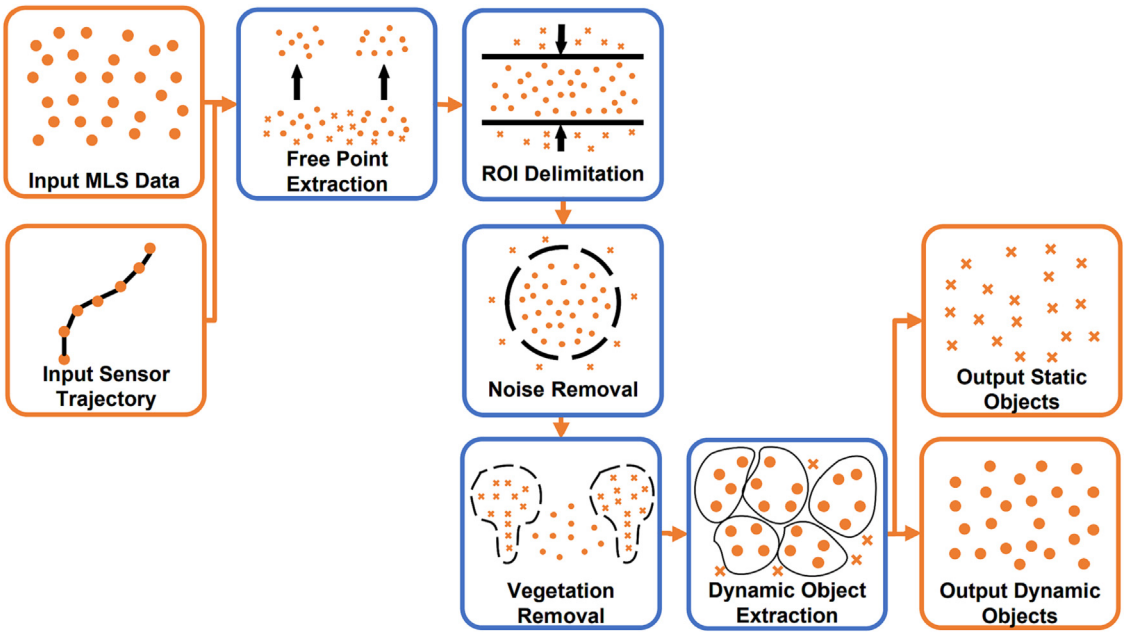

Fig. 2 Main workflow.

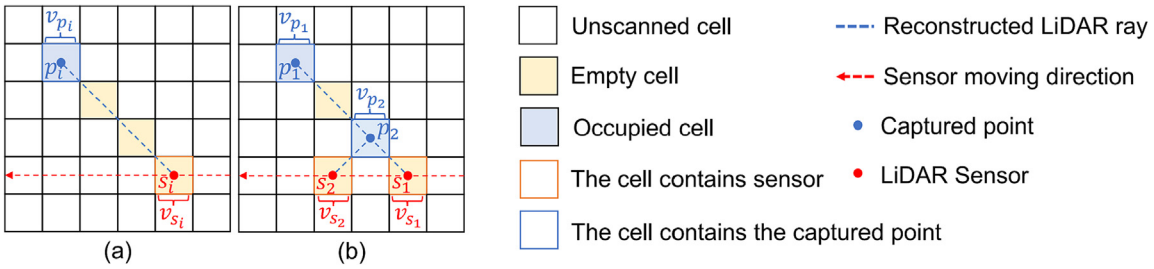

Fig. 3 Spatial classification based on ray tracing and the possible spatial conflict in Octomap.

mine the spatial type of each voxel cell v_i, but firstly calculates the penetration rate of v_i (i.e., the ratio of rays that pass through v_i to all rays that reach v_i) as its occupancy probability. Then given an occupancy probability threshold $thres_{occupied}$, any voxel space in Octomap with an occupancy probability greater than $thres_{occupied}$ is a free space. Points located in the free space are considered free points, which are potential dynamic points [26].

However, when Octomap handles MLS data with a massive number of points and large coverage, it has to generate a very huge voxel grid. During the process of adding new points to Octomap, the insertion speed is not fixed, but decreases continuously. Therefore, it usually need large computational effort and memory consumption for processing MLS data using Octomap, especially for the high-resolution grid.

Our solution is to split the entire MLS data into multiple subsets. Here we choose to split the MLS data into multiple data frames by timeline rather than into multiple blocks by space. Because splitting the point cloud by time dimension allows for fast matching of capture points with corresponding sensor positions to rapidly reconstruct LiDAR rays based on timestamps or point cloud generation order. Another reason is that the time-based segmentation makes the sensor trajectory located inside the corresponding scan region (Fig. 5.(b)), while the space-based segmentation will make part of the sensor trajectory points located outside the scan region (see the trajectories in the orange circles in Fig. 5.(b)). Therefore, when reconstructing the rays in Octomap, the former generates a voxel grid of approximate size to the axis-aligned bounding box of the scan region, while the latter generates a larger voxel grid due to the wider distribution of trajectory points, leading to more computation and memory cost.

The splitting solution is shown in Fig. 4: Given a continuous scan of MLS data from moment t_{start} to moment t_{end}, and a time interval t_{itvl}. The original MLS data is split into n data frames ($n = \left\lceil \frac{(t_{end} - t_{start})}{t_{itvl}} \right\rceil$). The start moment of the i-th data frame is $t_{start} + (i-1) \cdot t_{itvl}$. Each data frame is inserted into a separate Octomap with a resolution of $size_{voxel}$. However, the MLS points obtained from the start and end scan moments/-positions of each data frame (e.g., positions p_1 and p_2 in Fig. 5) are much less than the other scan moments/positions, which means the scans at these two positions are incomplete (Fig. 5. (b)) resulting in inaccurate occupancy probability estimates. Therefore, the points of two neighbouring data frames need to be partially introduced to enhance the start and end

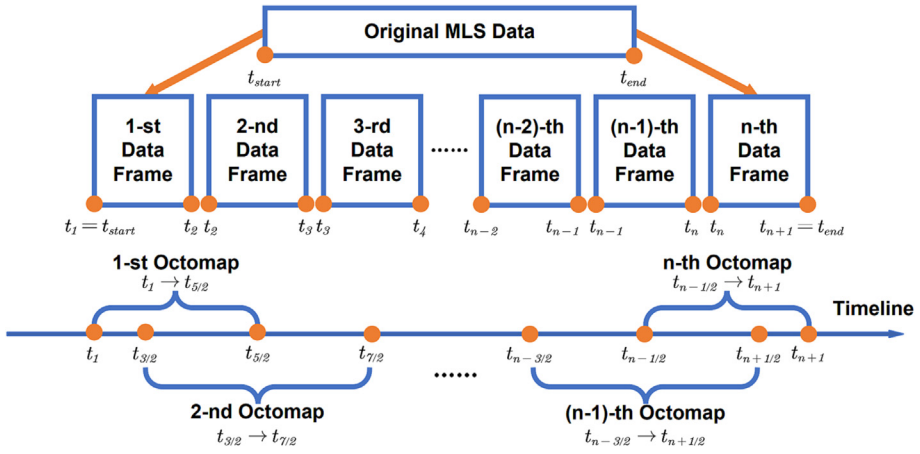

Fig. 4 Data frames segmentation.

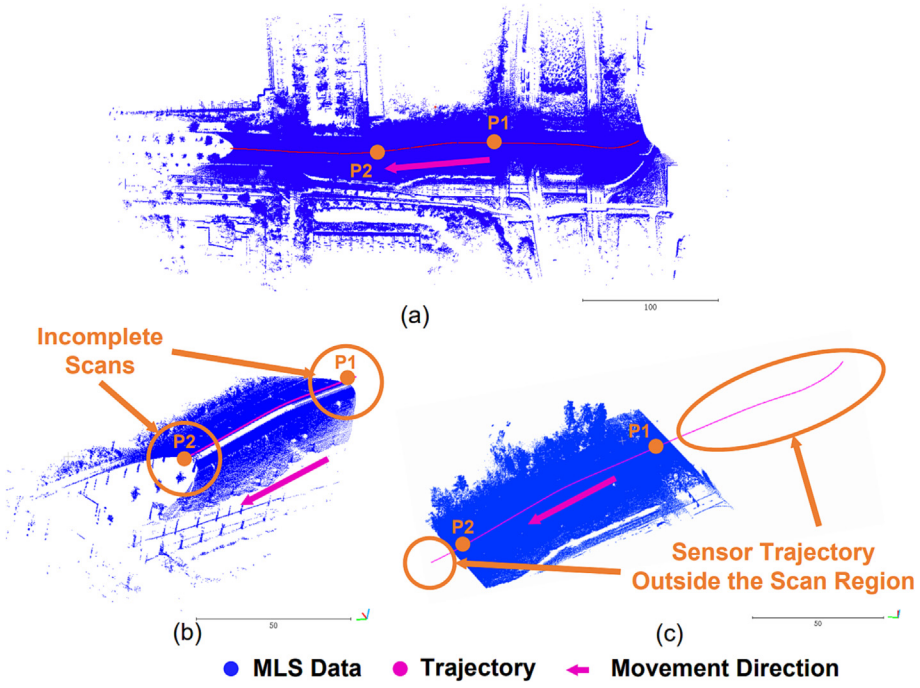

Fig. 5 A data frame based on time segmentation (b) and a block based on space segmentation (c) from the whole MLS data (a).

scan positions of the current data frame. So, the start and end moments of the i-th data frame are expanded as $t_{start} + (i - 3/2) \cdot t_{itvl}$ and $t_{start} + (i + 1/2) \cdot t_{itvl}$, respectively. This means all Octomaps are inserted with 2 data frames, except for the first and last Octomaps, which are only inserted with 1.5 data frames.

The next step is to extract the free points from each Octomap. Note that there is an overlap of $1/2 \cdot t_{itvl}$ between two neighbouring data frames (see the timeline in Fig. 4), which means their free points are also partially duplicated. Therefore, the extracted free points need to be de-redundant.

3.2. ROI delimitation

This study focuses only on land-based dynamic objects since they are the most relevant in MLS data, and thus a region of interest (ROI) is delimited using sensor trajectory, sensor mounting height, and local vehicle height restriction, to extract the free points near the ground surface for reducing the amount of calculation in subsequent steps. Here ROI is defined as the space between the height of the ground surface and the maximum allowable height of a large vehicle (excluding the ground).

The ground surface is a common lower boundary of ROI in dynamic object detection tasks [13,49,66]. The first reason is that the ground has very high point density and does not have any dynamic objects. So removing the ground reduces the computational cost of the subsequent step. Another reason is that the land-based dynamic objects are in contact with the ground. By removing the ground points, the objects are not connected by the ground surface, so they are more easily segmented into separate objects for detection tasks [3]. However,

most previous studies ignored the upper boundary of the ROI, resulting the upper part of some street-facing buildings and the crowns of some tall trees existing in the ROI. It hinders dynamic object detection and extraction. Since MLS data focuses on roads and their surroundings, vehicles are usually the largest dynamic objects in MLS data. The height of large vehicles is usually limited by the local form of regulations. Vehicles that exceed the height restriction cannot safely pass-through local transportation facilities such as tunnels [40]. Therefore, the height restriction of large vehicles is a reasonable upper boundary of the ROI in this study.

Fig. 6 shows how to obtain the height of the upper (h_{max_i}) and lower (h_{min_i}) boundaries of the ROI corresponding to each MLS sensor position s_i. The 3D coordinates $(x_{s_i}, y_{s_i}, h_{s_i})$ of s_i are obtained directly from the sensor trajectory data. h_{s_i} is the absolute height of s_i. Then given the sensor mounting height h_{sm}, the local lower boundary height of ROI (i.e., local ground height h_{min_i}) is calculated by $h_{min_i} = h_{s_i} - h_{sm}$. Next given the local vehicle height restriction h_{vr}, the local upper boundary of ROI (h_{max_i}) is obtained by $h_{max_i} = h_{min_i} + h_{vr}$. If the sensor captures a point p_i (3D coordinates: $x_{p_i}, y_{p_i}, h_{p_i}$) at position s_i and the ground height does not change within a certain range, the local lower boundary (ground) and upper boundary heights corresponding to p_i are also h_{min_i} and h_{max_i}. Thus, Eqn 1 determines whether the captured point p_i is in the ROI.

$$spacecategory = \begin{cases} ground : if h_{min_i} \geq h_{p_i} \\ ROI : h_{min_i} < h_{p_i} < h_{max_i} \\ high - altitude space : if h_{max_i} \leq h_{p_i} \end{cases} \quad (1)$$

It should be noted that the ROI delimitation must be done after the extraction of free points in this research. Although ground and high-altitude spaces do not contain dynamic objects, they still contain many LiDAR rays that are helpful for Octomap to calculate the occupancy probability of the target space more accurately. Therefore, they must be retained during the phase of the Octomap construction.

3.3. Noise removal

Although Octomap fully considers the effect of noise compared with other detection methods, some noise still inevitably remains in the extracted free points. It is undesirable to directly denoise free points base on their density because the overall point distribution of the MLS data is anisotropic and inhomogeneous [46] influenced by the scan mechanic and pattern of the MLS sensor [42]. In general, the returns of MLS sensor pulses decay as the distance to the sensor increases [34], which means that edge points in MLS data captured in outdoor open space are easily misidentified as noise due to their low density. Therefore, here noise and dynamic points in free points are distinguished by comparing the local density in the free point set and the original MLS data (i.e., free point rate, see Fig. 7). For each free point p_i, its free point rate $rate_{fp_i} = density_{fp_i} / density_{MLS_i}$. Where $density_{fp_i}$ is the local density in the free point set within radius r_{ns} and $density_{MLS_i}$ is the local density in the original MLS data within the same radius. Given the free-point rate threshold $thres_{fp}$, if $thres_{fp} > rate_{fp_i}$, the free point p_i is noise, otherwise it is a non-noisy point.

3.4. Vegetation removal

Vegetation, such as grasslands and tree crowns, are easily misidentified as free points in Octomap due to their sparse and non-rigid structure. Therefore, it is necessary to remove vegetation points before extracting dynamic objects from free points. Many related studies have focused on using LiDAR data for vegetation extraction in urban environments [22]. Several previous studies have demonstrated that the number of returned LiDAR rays and vegetation are closely related [4,14,24]. The sparse and non-rigid structure of the vegetation easily produces multiple returns in the LiDAR data. Specifically, a laser pulse may half-hit a leaf or branch and cause multiple returns.

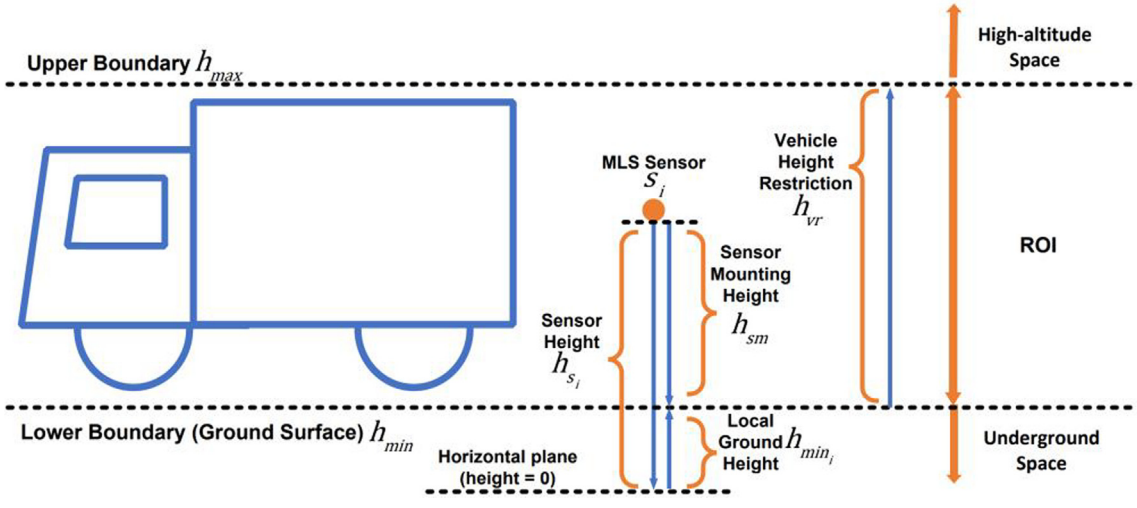

Fig. 6 ROI delimitation.

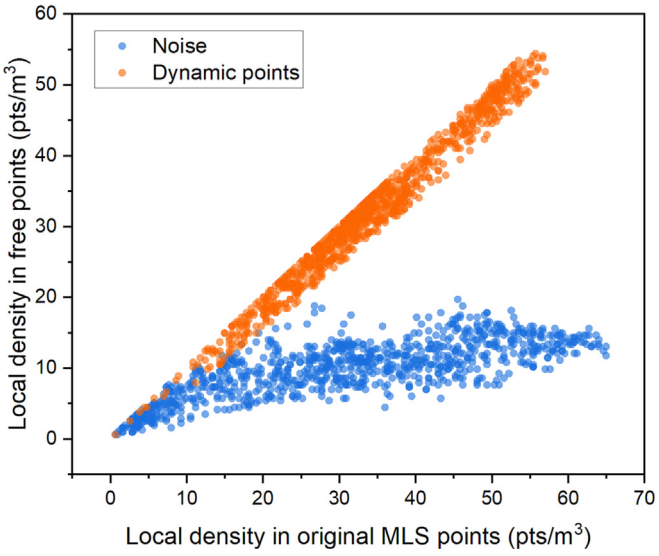

Fig. 7 Relationship between local density in original MLS points and local density in free point set for 1,000 dynamic points and 1,000 noisy points randomly sampled from the experimental data (r = 1 m).

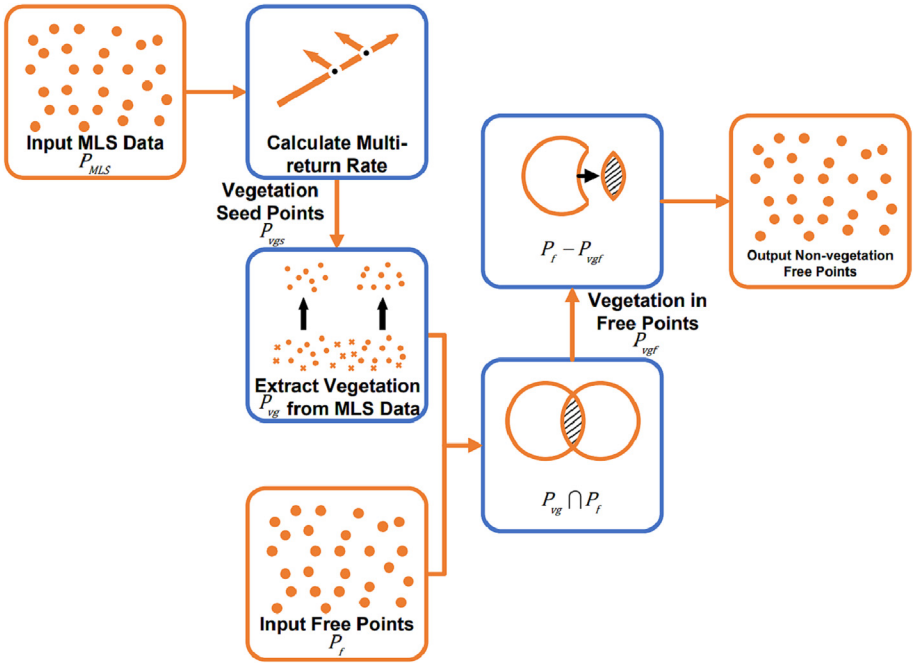

Fig. 8 The workflow of vegetation removal.

So, vegetation points are detected and removed from following steps (Fig. 8): Firstly, for each original MLS point p_i, given the number of its multi-returned neighbours (num_{nbmr_i}, i.e., the point in the neighbourhood that generates multiple return pulses) and the number of all its neighbours (num_{nb_i}) in the neighbourhood with radius r_{mr}, its multi-return rate $rate_{mr_i}$ is obtained from $rate_{mr_i} = num_{nbmr_i}/num_{nb_i}$. If $rate_{mr_i}$ is greater than the pre-set multi-return rate threshold $thres_{mr}$, p_i will be marked as a vegetation seed point. Then all vegetation points (P_{vg}) in the original MLS data (P_{MLS}) are extracted with obtained vegetation seed points (P_{vgs}) by a region growing algorithm with radius search r_{oi}. Vegetation points are not directly extracted from free points because the input MLS data has higher density than free points, vegetation cab be clustered correctly. Then, free points (P_f) intersecting with vegetation points (P_{vg}) are removed.

3.5. Dynamic object extraction

The last step is extracting dynamic objects from the original MLS data using filtered free points as seed points with a region growing algorithm with search radius r_{oi}.

Although the previous noise and vegetation filtering operations have removed most of the vegetation and noise points

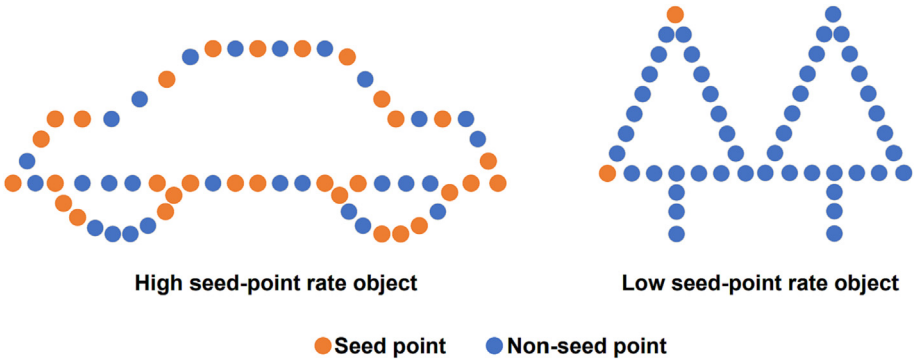

Fig. 9 Objects with the high seed-point rate (left) and low seed-point rate (right).

from the free points, a very small amount of non-dynamic points may inevitably remain in the filtered free points, which lead to some fake dynamic objects with a very small proportion of seed points. These fake dynamic objects are mainly caused by a small number of non-dynamic points falling near the vegetation. These non-dynamic points connect to surrounding grasslands or urban forests and misidentify them as large dynamic objects. For this type of fake dynamic object, the seed points are only a small fraction of all points. So, for each valid dynamic object candidate o_i, its proportion of seed points $rate_{sp_i} = num_{sp_i}/num_{op_i}$ (i.e., seed-point rate. num_{sp_i} is the number of seed points in o_i and num_{op_i} is the total point number of o_i) should be larger than the pre-set seed-point rate threshold $thres_{sp}$ (Fig. 9). Finally, the real dynamic objects are obtained after removing these fake dynamic objects.

4. Experimental results and discussion

The experimental results and performance of our proposed methods are evaluated at four case sites in Delft, the Netherlands. First, the evaluation dataset, four case sites, and all involved parameter values are introduced in Section 4.1 and Section 4.2. Next, Section 4.3 assesses the experimental accuracies. Then Section 4.4 discusses the factors influencing the performance of our method. Finally, Section 4.5 lists the running time and memory consumption of the proposed method and compares it with the original Octomap.

4.1. Experimental dataset and case sites

The MLS data were collected by CycloMedia Technology with a vehicle-mounted Velodyne's HDL-32E LiDAR sensor, in Delft, the Netherlands, in July 2021. The MLS data includes information such as GNSS time and the number of returns, in addition to 3D spatial coordinates. The corresponding sensor trajectories were obtained from GNSS and IMU integrated in the MLS platform.

Four case sites were selected for this research (see Fig. 10 and Table 1 for their exact positions). They were all located at road junctions with had bicycle lanes and sidewalks. The road network at the road junctions is a complex environment influenced by traffic signals, so dynamic objects with different speeds and trajectories can be observed in these areas. Table 2 lists detailed information about the captured MLS data and sensor trajectories captured and Fig. 11 shows the MLS scan data, corresponding sensor trajectory data, and satellite images for the four case sites. Each case site was scanned for 10 s. Since each case site varies regarding the driving speed of the data acquisition vehicle, the surrounding static environment, and the real-time status of dynamic objects while being scanned, this results in differences in the point cloud size, point cloud density, dynamic point percentage, and scanning range of the four captured data.

4.2. Parameters

Table 3 lists all parameter values involved in the proposed method. A small t_{itvl} allows for lower computational and memory costs when processing each Octomap, but too small t_{itvl} leaves insufficient LiDAR rays within each Octomap. Thus, here t_{itvl} is set to 0.75 sec (14 data frames). Values for $size_{voxel}$ (0.2 m) and $thres_{occupied}$ (0.7) refer to the work of Gehrung et al. [20]. The upper (h_{vr}) and lower (h_{sm}) boundaries of ROI are obtained from the large vehicle height limit in the Netherlands and the MLS platform of CycloMedia Technology. Multi-return rate threshold $thres_{mr}$ is based on the values used in some previous research [55,61]. Based on the results presented in Fig. 12, 0.84 is a appropriate free-point rate threshold in this study, which removes most noise while preserving the majority of dynamic points. The two constraints (num_{min} and $thres_{sp}$) used to filter the valid dynamic objects should avoid being set too small, otherwise they will fail to recognize some small objects, such as cyclists, and sparse objects far from the sensor. The search radius is a key parameter for fixed radius neighbourhood search. Too large or too small search radius affects the intermediate and final results of our proposed method. Therefore, the radiuses (r_{ns} and r_{mr}) for free-point rate and multi-return rate are moderately set to 1 m, while the radius (r_{oi}) for object individualization is 0.5 m.

4.3. Accuracy assessment

Dynamic object detection and extraction is essentially a point cloud binary classification issue. Therefore, we first manually labelled the dynamic and static objects on the experimental data as ground truth (Fig. 13). Then we use confusion matri-

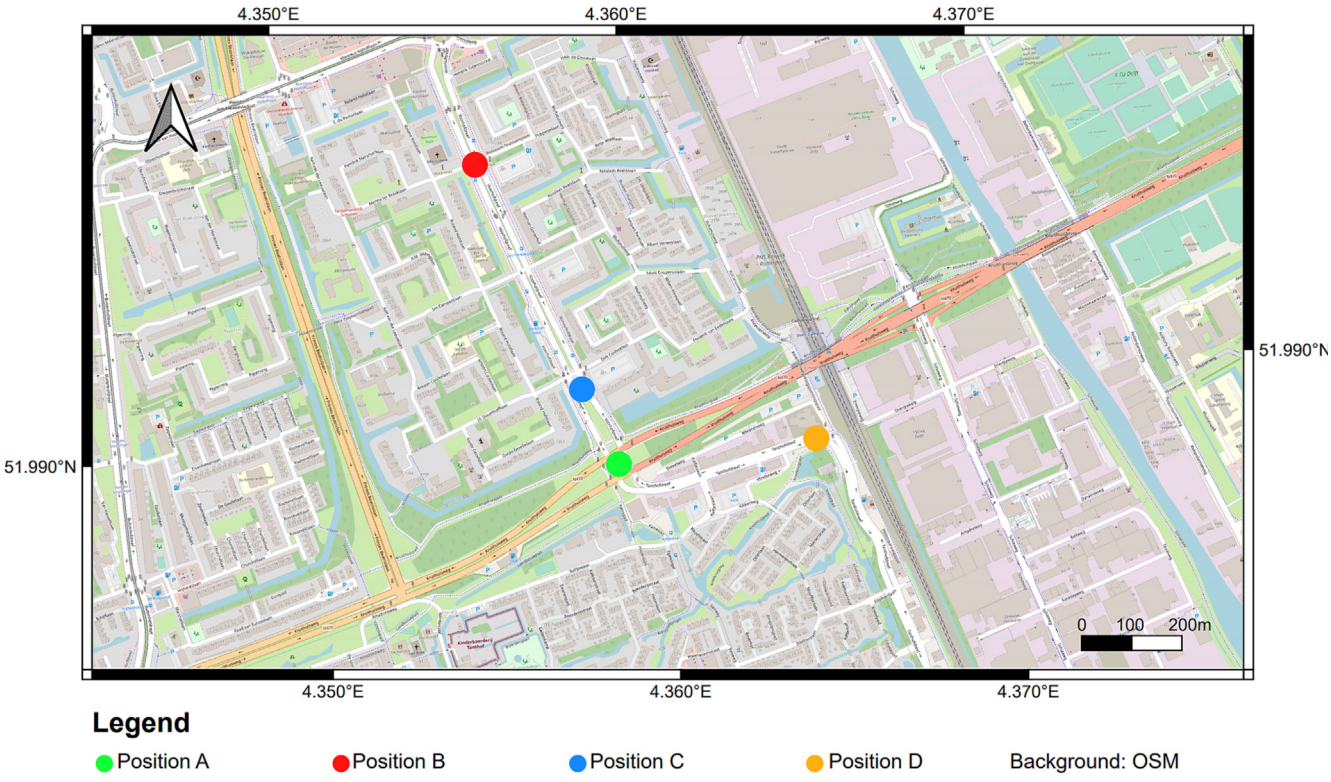

Fig. 10 Positions of the four case sites.

ces, user's accuracy, producer's accuracy, and overall accuracy (Table 4) to evaluate our proposed method and compare these indicators with those of the original Octomap method [26].

Table 5 and Table 6 show the confusion matrices and three accuracy assessment indicators of the four case sites based on our proposed method and original Octomap. The weight for overall accuracy is the number of points captured in the four case sites. The weight for user's/producer's accuracy of dynamic/static points is the number of dynamic/static points in each detection result/ground truth. In our method, the four case sites differ in their respective detection accuracies due to their different proportions of the dynamic objects, directions of motion, and distances to the sensors. However, overall high weighted average SUA (99.97%) and SPA (99.89%) values indicate that almost all static objects are detected correctly. The value of OA is also very high (99.85%) due to the high percentage of static objects in the experimental data. For the accuracies of dynamic objects, DUA and DPA are 84.98% and 94.91%, respectively. This means most dynamic objects in the MLS data for the four case sites are successfully detected and extracted. Our method performs better on DUA compared to the original Octomap and the other indicators are basically the same as the original Octomap method.

Table 1 Full name of the four case sites.

Case Site	Full Name of the Case Site
Position A	Voorhofdreef, Tanthofdreef and Kruithuisweg
Position B	Voorhofdreef, Menno Ter Braaklaan, and Bosboom-Toussaintplein
Position C	Voorhofdreef, J.J. Slauerhofflaan, and Frederik van Eedenlaan
Position D	Tanthofdreef and Forensenweg

4.4. Discussion

In previous related research, the detection of low-speed objects is considered more challenging than high-speed objects [15]. However, the speed of objects does not significantly affect the detection in our experimental results. Most low-speed

Table 2 Details of the MLS data at the four case sites and the corresponding sensor trajectories.

Case Site	Scan Time (sec)	Point Size	Dynamic Point Size	Percentage of Dynamic Points	Mean Point Density r = 1 (pt/m^3)	Trajectory Length (m)	Mean Sensor Movement Velocity (m/s)
Position A	10	4,547,462	68,820	1.51%	1083.86	96.11	9.61
Position B	10	4,616,356	12,931	0.28%	695.55	122.51	12.25
Position C	10	4,666,430	35,763	0.77%	1026.62	83.87	8.39
Position D	10	4,684,840	9164	0.20%	997.86	108.94	10.89

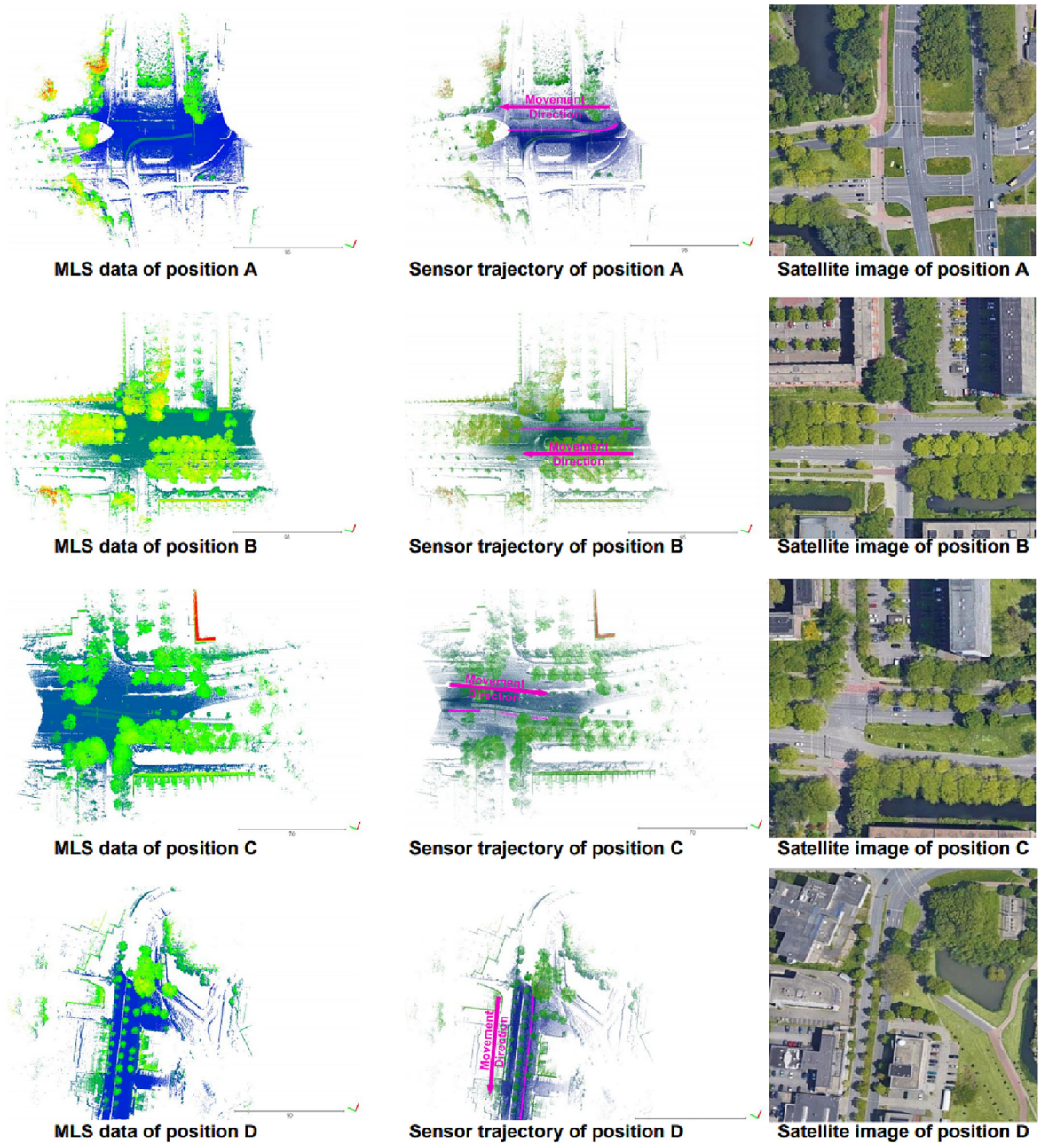

Fig. 11 MLS data (left column) rendered in height, corresponding sensor trajectories (middle column) colorized in the purple, and satellite images from Google Map (right column) of the four case sites. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

objects such as vehicles that are braking (see the upper left corner of Fig. 14.(a)) and moving bicycles (Fig. 14.(b-c)) are correctly detected. Many successfully detected bicycles also demonstrate the applicability of our method to small objects.

We evaluate the effect of the motion direction of the dynamic object on our detection results. Fig. 15 illustrates several objects that move in directions different from the MLS sensor. These dynamic objects are all successfully detected despite their different motion directions. Above results indicate that our method works well regardless of the shape of the dynamic objects, nor of their movement relative to the MLS sensor.

The false dynamics in the detection results are mainly caused by mis-detected vegetation (Fig. 16.(a-b)). The secondary cause is the performance of the ground filtering method proposed in Section 3.2, which assumes that the ground is a flat surface and does not vary significantly in height within a certain range. But some surfaces in the real world do not conform to this assumption, such as the two ground areas in position A. These two ground areas are not successfully detected as static objects due to their uneven surfaces (Fig. 16.(e)). However, only part of the points in these two ground areas are detected as dynamic points and do not leave significant voids in the ground. Therefore, the ground

Table 3 Values of implementation parameters.

Parameter	Description	Value
t_{itvl}	Time interval for data frame segmentation	0.75 sec
$size_{voxel}$	Octomap voxel size	0.2 m
$thres_{occupied}$	Occupancy probability threshold	0.7
h_{sm}	Sensor mounting height	2 m
h_{vr}	Height restriction of large vehicles	4 m
$thres_{fp}$	Free-point rate threshold	0.84
r_{ns}	Search radius used to calculate the free-point rate	1 m
$thres_{mr}$	Multi-return rate threshold	0.3
r_{mr}	Search radius used to calculate the multi-return rate	1 m
r_{oi}.	Search radius for object individualization	0.5 m
num_{min}	Minimum point number limit for dynamic objects	15
$thres_{sp}$	Threshold of seed point proportion	0.03

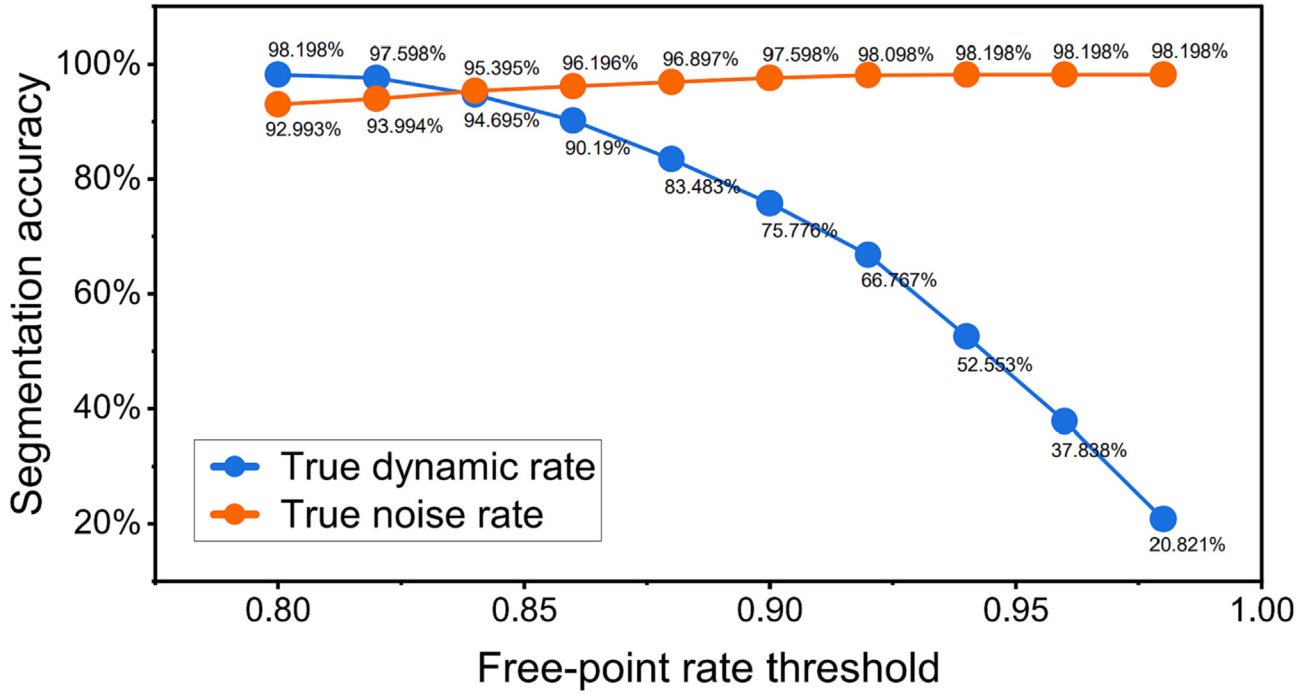

Fig. 12 Segmentation accuracies obtained by applying different free-point rate thresholds to the dynamic points and noise in Fig. 7.

connectivity is still intact (Fig. 16.(f)). In addition, other objects result in false dynamic including remnant buildings (Fig. 16.(d)) and pole-like objects such as streetlights, traffic lights, and traffic signs (Fig. 16.(c)).

The main reason for the false static is the distance from the dynamic object to the MLS sensor. When the dynamic object is too far away from the MLS sensor, it usually has low point density, which resulting in incomplete detection (Fig. 17). But this can be solved by adding previous or next data frames to improve the point density of the object if it is in the edge area in front and behind the sensor. For the other points far from the MLS sensor (i.e., located on the left and right side of the sensor trajectory), due to their low completeness and accuracy, they are not useful for most applications. The detection accuracy is higher if the dynamic objects are closer to the MLS sensor. The main reason for the high detection accuracy in position C is that most dynamic objects in this area are very close to the MLS sensor and have similar motion trajectories to the sensor.

4.5. Running time and memory consumption

The proposed methodology is based on a C++ implementation, which is run on a 64-bit Windows operation system and an AMD Ryzen-9 3.30 GHz CPU with 16 GB RAM. We set Octomaps to update their occupancy probabilities every insertion of 500 LiDAR rays, and then record the running times of our proposed method for the four case sites in Fig. 18. Our method uses the built-in *std::future* and *std::async*

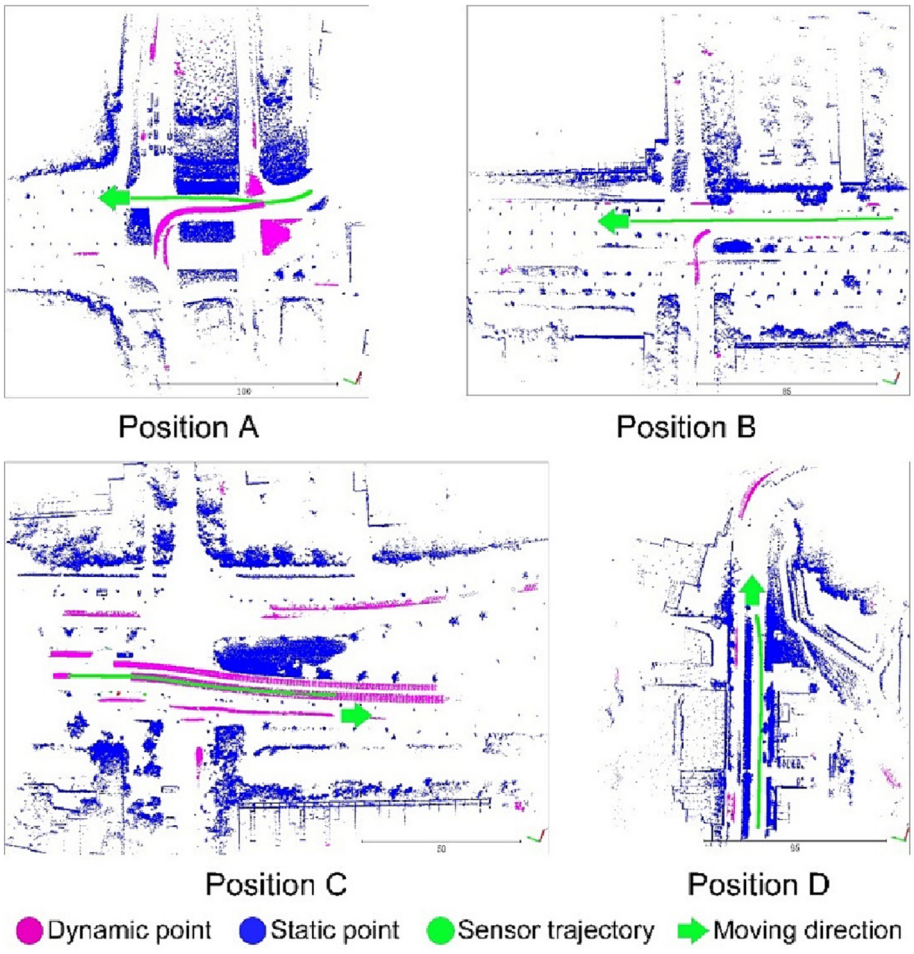

Fig. 13 Point cloud coloured according to detection results.

Table 4 Descriptions and equations of user's accuracy, producer's accuracy, and overall accuracy.

Indicator	Description	Equation
Dynamic User's Accuracy (DUA)	Percentage of dynamic points from the experimental result that are correctly detected.	$\frac{TD}{TD+FD} \cdot 100\%$
Static User's Accuracy (SUA)	Percentage of static points from the experimental result that are correctly detected.	$\frac{TS}{FS+TS} \cdot 100\%$
Dynamic Producer's Accuracy (DPA)	Percentage of dynamic points from the ground truth that are correctly detected.	$\frac{TD}{TD+FS} \cdot 100\%$
Static Producer's Accuracy (SPA)	Percentage of static points from the ground truth that are correctly detected.	$\frac{TS}{FD+TS} \cdot 100\%$
Overall Accuracy (OA)	Overall percentage of dynamic and static points that are correctly detected.	$\frac{TD+TS}{TD+FD+FS+TS} \cdot 100\%$

parallel computation functions in C++ to achieve acceleration. The average computational efficiency is increased by 43.47% with 2 threads enabled and 107.30% with 4 threads enabled. The efficiency improvement from parallel computing is lower than the theoretical value, mainly because each data frame contains different number of points and 3D geometries, which is especially noticeable between the centre and edge data frames. This results in a different amount of data to be processed by each thread. Compared to the original Octomap method using the same parameters, our method with 1 thread enabled accelerates on average by 18.27% and the maximum memory consumption is reduced to 39.40% (Fig. 19) in the

Table 5 Confusion matrix for four case sites using original Octomap.

Predicted\Reference		Proposed Method		Original Octomap	
		Dynamic Points	Static Points	Dynamic Points	Static Points
A	Dynamic Points	68,633	13,679	67,817	15,045
	Static Points	187	4,464,963	1003	4,463,597
B	Dynamic Points	11,002	3597	12,300	6083
	Static Points	1665	4,600,092	367	4,597,606
C	Dynamic Points	33,529	1978	33,817	4927
	Static Points	2234	4,628,689	1946	4,625,740
D	Dynamic Points	6813	1951	8969	1873
	Static Points	2351	4,673,725	195	4,673,803

Table 6 User's accuracies, producer's accuracies, and overall accuracies of the four case sites using our proposed method (PM) and original Octomap (OO).

Case Site	DUA		SUA		DPA		SPA		OA	
	PM	OO	PM	OO	PM	OO	PM	OO	PM	OO
A	83.38%	81.84%	100.00%	99.98%	99.73%	98.54%	99.70%	99.66%	99.70%	99.65%
B	75.36%	66.91%	99.96%	99.99%	86.86%	97.10%	99.92%	99.87%	99.89%	99.86%
C	94.43%	87.28%	99.95%	99.96%	93.75%	94.56%	99.96%	99.89%	99.91%	99.85%
D	77.74%	82.73%	99.95%	100.00%	74.35%	97.87%	99.96%	99.96%	99.91%	99.96%
Weighted Average	84.98%	81.48%	99.97%	99.98%	94.91%	97.22%	99.89%	99.85%	99.85%	99.83%

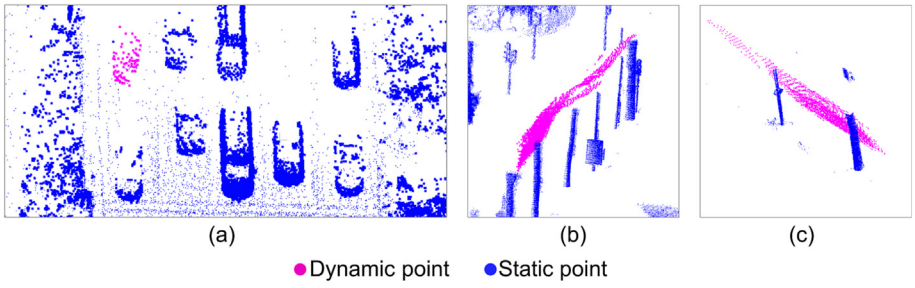

Fig. 14 Detected braking vehicle (a) and moving bicycles (b-c).

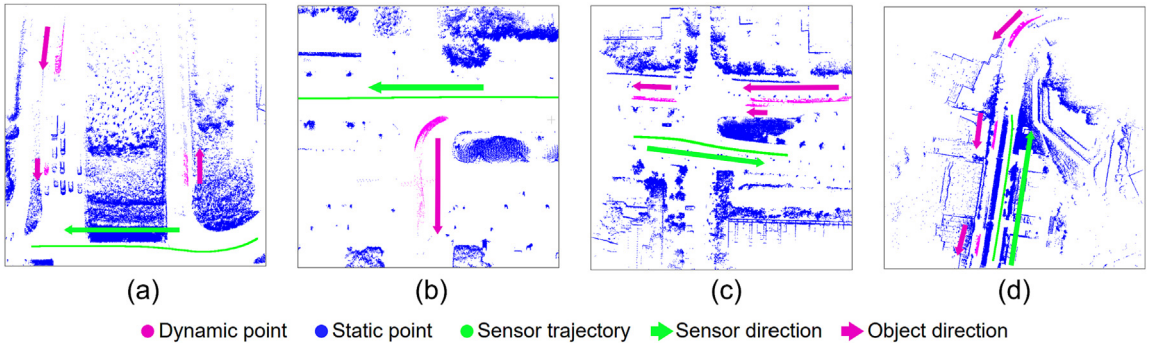

Fig. 15 Objects move in the direction perpendicular (a-b) or opposite (c-d) to the direction of the MLS sensor.

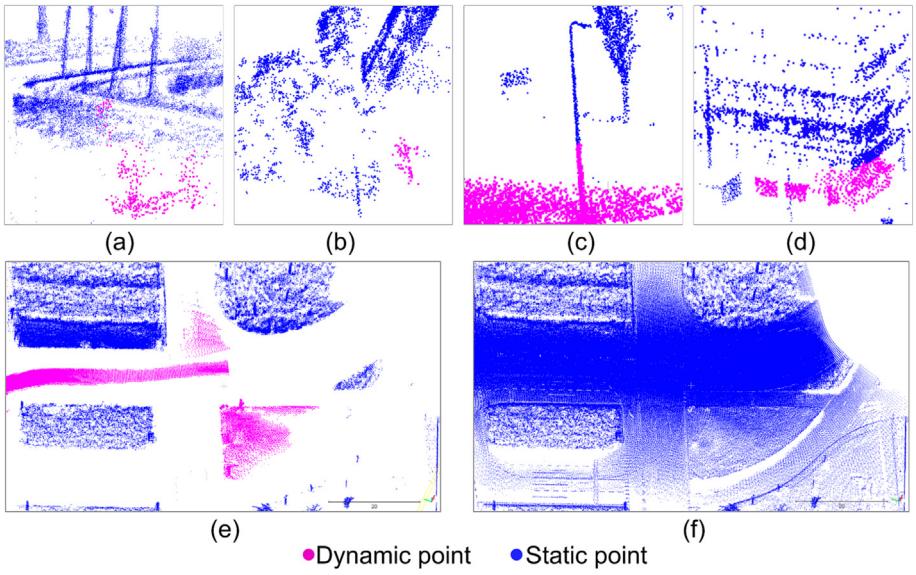

Fig. 16 Mis-detected vegetation (a-b), pole-like object (c), remnant building (d), ground points (e) and their corresponding ground surface (f).

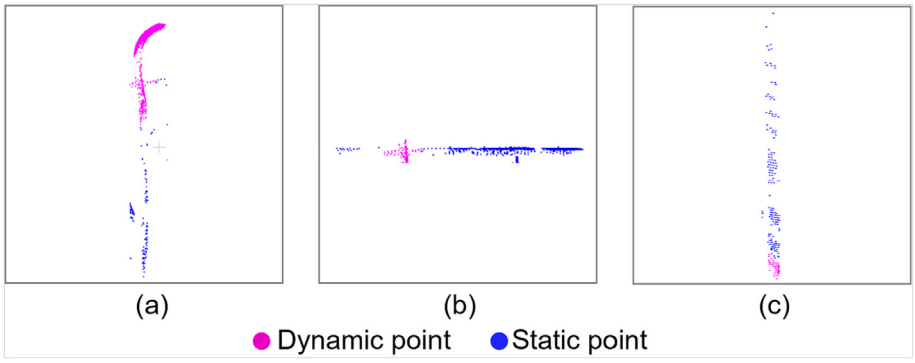

Fig. 17 Top view of sparse false static objects, which are mainly incompletely detected vehicles (a-b) and the low point density object (c).

voxel grid construction and free point extraction. The above improvements are mainly achieved by splitting the huge voxel grid in the original Octomap into several smaller voxel grids.

5. Conclusions

We propose a method for detecting and removing dynamic objects in MLS data based on a data frame aware optimized Octomap. The Octomap is decomposed into multiple smaller Octomaps to improve the free point extraction performance. Then most of the non-dynamic free points are removed by delineating a smaller ROI as well as calculating the free-point rate and the multi-return rate. Finally, dynamic objects are detected and removed by fixed-radius neighbourhood search. The main innovation of our proposed method is to introduce the data frame and the ROI upper boundary into the Octomap-based method to optimize performance and to make it more compatible with parallel computing.

In experimental results, our method correctly detects 84.98% of dynamic objects and shows good performance regardless of the shape of the dynamic objects, and their velocity and motion relative to the MLS sensor. Without decreasing the detection accuracy, our method improves the efficiency by 18.27% over the original Octomap and can use parallel computing to achieve further acceleration. The maximum memory consumption is only 39.40% of the original Octomap.

In the future, our method will be integrated with GPU acceleration, non-uniform grid structures, and database management systems (DBMS) for further acceleration. Classification algorithms will also be implemented to reduce de number of misclassified objects.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

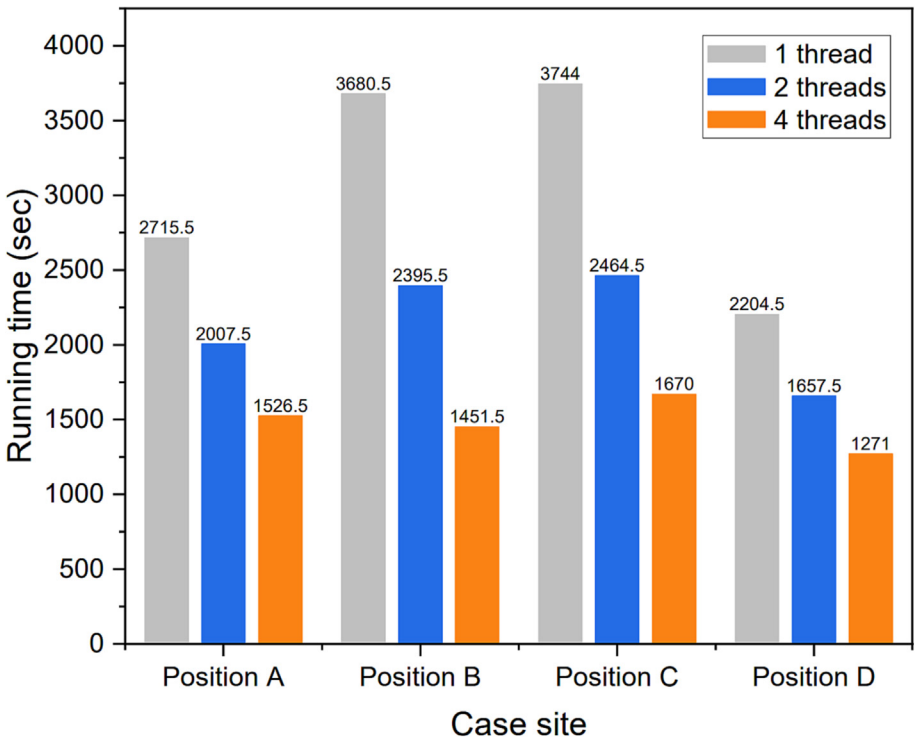

Fig. 18 Running time of our proposed method using 1, 2, and 4 thread(s) in four case sites.

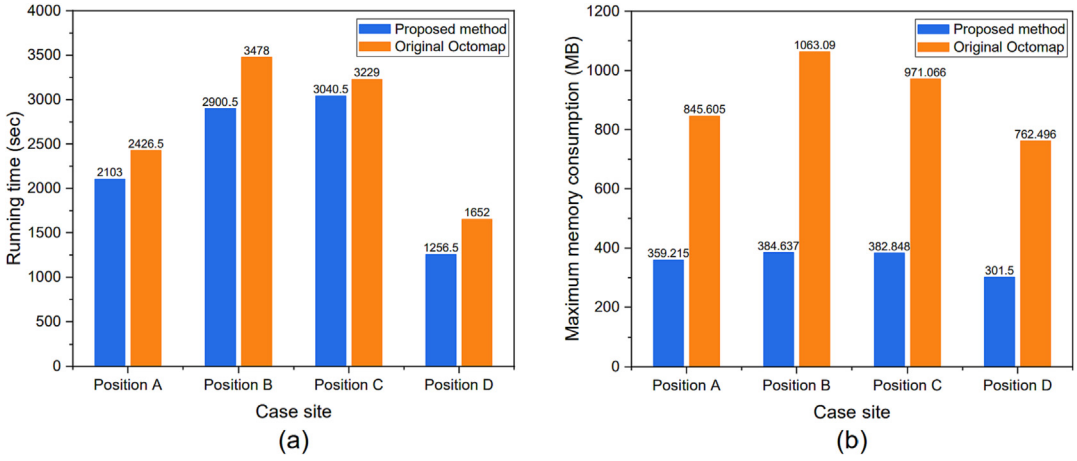

Fig. 19 Running time (a) and maximum memory consumption (b) of voxel grid generation and free point extraction using our proposed method and original Octomap.

Acknowledgements

Authors would like to thank the Xunta de Galicia given through human resources grant (ED481B-2019-061)

References

[1] P. Anderson-Sprecher, R. Simmons, D. Huber, Background subtraction and accessibility analysis in evidence grids, IEEE Int. Conf. Robot. Autom. 2011 (2011) 3104–3110, https://doi.org/10.1109/ICRA.2011.5980428.

[2] G. Arisholm, T. Skauli, S. Landrø, Combined range ambiguity resolution and noise reduction in lidar signal processing, Opt. Eng. 57 (07) (2018) 1, https://doi.org/10.1117/1.OE.57.7.073103.

[3] M. Arora, L. Wiesmann, X. Chen, C. Stachniss, Mapping the static parts of dynamic scenes from 3D LiDAR point clouds exploiting ground segmentation, Eur. Conf. Mobile Robots (ECMR) 2021 (2021) 1–6, https://doi.org/10.1109/ECMR50962.2021.9568799.

[4] J. Balado, P. Arias, L. Díaz-Vilariño, L.M. González-deSantos, Automatic CORINE land cover classification from airborne LIDAR data, Procedia Comput. Sci. 126 (2018) 186–194, https://doi.org/10.1016/j.procs.2018.07.222.

[5] J. Balado, L. Díaz-Vilariño, P. Arias, H. Lorenzo, Point clouds for direct pedestrian pathfinding in urban environments, ISPRS J. Photogramm. Remote Sens. 148 (2019) 184–196, https://doi.org/10.1016/j.isprsjprs.2019.01.004.
[6] J. Balado, E. González, E. Verbree, L. Díaz-Vilariño, H. Lorenzo, Automatic detection and characterization of ground occlusions in urban point clouds from mobile laser scanning data, ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, VI-4/W1-2020, 2020, 13–20. https://doi.org/10.5194/isprs-annals-VI-4-W1-2020-13-2020.
[7] M. Barbarella, A. Di Benedetto, M. Fiani, A method for obtaining a DEM with curved abscissa from MLS data for linear infrastructure survey design, Remote Sens. (Basel) 14 (4) (2022) Article 4, https://doi.org/10.3390/rs14040889.
[8] A. Börcs, C. Benedek, A marked point process model for vehicle detection in aerial lidar point clouds, ISPRS Annals Photogram., Remote Sens. Spatial Inf. Sci. I–3 (2012) 93–98, https://doi.org/10.5194/isprsannals-I-3-93-2012.
[9] C. Buerkle, F. Oboril, J. Jarquin, K.-U. Scholl, Efficient dynamic occupancy grid mapping using non-uniform cell representation, IEEE Intell. Vehic. Symp. (IV) 2020 (2020) 1629–1634, https://doi.org/10.1109/IV47402.2020.9304571.
[10] X. Chen, A. Milioto, E. Palazzolo, P. Giguère, J. Behley, C. Stachniss, SuMa + +: Efficient LiDAR-based Semantic SLAM, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) 2019 (2019) 4530–4537, https://doi.org/10.1109/IROS40897.2019.8967704.
[11] Y. Chen, S. Sun, H. Yin, M.H. Ang, Exploring the Effect of 3D Object Removal using Deep Learning for LiDAR-based Mapping and Long-term Vehicular Localization, in: 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), 2022, pp. 1730–1735, https://doi.org/10.1109/ITSC55140.2022.9921969.
[12] J. Cheng, Z. Xiang, T. Cao, J. Liu, Robust vehicle detection using 3D Lidar under complex urban environment, IEEE International Conference on Robotics and Automation (ICRA) 2014 (2014) 691–696, https://doi.org/10.1109/ICRA.2014.6906929.
[13] Choi, J., Ulbrich, S., Lichte, B., & Maurer, M. (2013). Multi-Target Tracking using a 3D-Lidar sensor for autonomous vehicles. 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013), 881–886. https://doi.org/10.1109/ITSC.2013.6728343.
[14] M. Dalponte, N.C. Coops, L. Bruzzone, D. Gianelle, Analysis on the use of multiple returns LiDAR data for the estimation of tree stems volume, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 2 (4) (2009) 310–318, https://doi.org/10.1109/JSTARS.2009.2037523.
[15] A. Dewan, T. Caselitz, G.D. Tipaldi, W. Burgard, Motion-based detection and tracking in 3D LiDAR scans, IEEE Int. Conf. Robot. Autom. (ICRA) 2016 (2016) 4508–4513, https://doi.org/10.1109/ICRA.2016.7487649.
[16] P. Ding, Z. Wang, 3D LiDAR point cloud loop detection based on dynamic object removal, IEEE International Conference on Real-Time Computing and Robotics (RCAR) 2021 (2021) 980–985, https://doi.org/10.1109/RCAR52367.2021.9517428.
[17] D. Duberg, P. Jensfelt, UFOMap: An Efficient Probabilistic 3D Mapping Framework That Embraces the Unknown, IEEE Rob. Autom. Lett. 5 (4) (2020) 6411–6418, https://doi.org/10.1109/LRA.2020.3013861.
[18] Y. Endo, E. Javanmardi, S. Kamijo, Analysis of occlusion effects for map-based self-localization in urban areas, Sensors 21 (15) (2021) Article 15, https://doi.org/10.3390/s21155196.
[19] M.A. Fischler, R.C. Bolles, Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, Commun. ACM 24 (6) (1981) 381–395, https://doi.org/10.1145/358669.358692.
[20] J. Gehrung, M. Hebel, M. Arens, U. Stilla, An approach to extract moving objects from mls data using a volumetric background representation, ISPRS Annals Photogram., Remote Sens. Spatial Inf. Sci. IV-1/W1 (2017) 107–114, https://doi.org/10.5194/isprs-annals-IV-1-W1-107-2017.
[21] A. Geiger, P. Lenz, C. Stiller, R. Urtasun, Vision meets robotics: The KITTI dataset, Int. J. Robot. Res. 32 (11) (2013) 1231–1237, https://doi.org/10.1177/0278364913491297.
[22] Q. Guo, Y. Su, T. Hu, H. Guan, S. Jin, J. Zhang, X. Zhao, K. Xu, D. Wei, M. Kelly, N.C. Coops, Lidar boosts 3d ecological observations and modelings: a review and perspective, IEEE Geosci. Remote Sens. Mag. 9 (1) (2021) 232–257, https://doi.org/10.1109/MGRS.2020.3032713.
[23] Z. Guo, B. Cai, W. Jiang, J. Wang, Feature-based detection and classification of moving objects using LiDAR sensor, IET Intel. Transport Syst. 13 (7) (2019) 1088–1096, https://doi.org/10.1049/iet-its.2018.5291.
[24] A. Gupta, J. Byrne, D. Moloney, S. Watson, H. Yin, Tree Annotations in LiDAR Data Using Point Densities and Convolutional Neural Networks, IEEE Trans. Geosci. Remote Sens. 58 (2) (2020) 971–981, https://doi.org/10.1109/TGRS.2019.2942201.
[25] A. Hermann, F. Drews, J. Bauer, S. Klemm, A. Roennau, R. Dillmann, Unified GPU voxel collision detection for mobile manipulation planning, IEEE/RSJ Int. Conf. Intell. Robots Syst. 2014 (2014) 4154–4160, https://doi.org/10.1109/IROS.2014.6943148.
[26] A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss, W. Burgard, OctoMap: An efficient probabilistic 3D mapping framework based on octrees, Auton. Robot. 34 (3) (2013) 189–206, https://doi.org/10.1007/s10514-012-9321-0.
[27] R. Huang, W. Zhang, A. Kundu, C. Pantofaru, D.A. Ross, T. Funkhouser, A. Fathi, An LSTM Approach to Temporal 3D Object Detection in LiDAR Point Clouds, in: A. Vedaldi, H. Bischof, T. Brox, & J.-M. Frahm (Eds.), Computer Vision – ECCV 2020, 2020, pp. 266–282. Springer International Publishing. https://doi.org/10.1007/978-3-030-58523-5_16.
[28] H. Iqbal, D. Campo, P. Marin-Plaza, L. Marcenaro, D.M. Gómez, C. Regazzoni, Modeling Perception in Autonomous Vehicles via 3D Convolutional Representations on LiDAR, IEEE Trans. Intell. Transp. Syst. 1–12 (2021), https://doi.org/10.1109/TITS.2021.3130974.
[29] G. Kim, A. Kim, Remove, then revert: static point cloud map construction using multiresolution range images, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2020 (2020) 10758–10765, https://doi.org/10.1109/IROS45743.2020.9340856.
[30] B.R. Kiran, L. Roldão, B. Irastorza, R. Verastegui, S. Süss, S. Yogamani, V. Talpaert, A. Lepoutre, G. Trehard, Real-Time Dynamic Object Detection for Autonomous Driving Using Prior 3D-Maps, in: L. Leal-Taixé, S. Roth (Eds.), Computer Vision –, Vol. 11133, Springer International Publishing, 2019, pp. 567–582, https://doi.org/10.1007/978-3-030-11021-5_35, ECCV 2018 Workshops.
[31] J. Ku, M. Mozifian, J. Lee, A. Harakeh, S.L. Waslander, Joint 3D proposal generation and object detection from view aggregation, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2018 (2018) 1–8, https://doi.org/10.1109/IROS.2018.8594049.
[32] H. Lim, S. Hwang, H. Myung, ERASOR: egocentric ratio of pseudo occupancy-based dynamic object removal for static 3d point cloud map building, IEEE Rob. Autom. Lett. 6 (2) (2021) 2272–2279, https://doi.org/10.1109/LRA.2021.3061363.
[33] Z. Lin, M. Hashimoto, K. Takigawa, K. Takahashi, Vehicle and Pedestrian Recognition Using Multilayer Lidar based on Support Vector Machine, in: 2018 25th International

Conference on Mechatronics and Machine Vision in Practice (M2VIP), 2018, 1–6. https://doi.org/10.1109/M2VIP.2018.8600877.
[34] J.C. Lock, F. Camara, C. Fox, EMap: Real-Time Terrain Estimation, in: S. Pacheco-Gutierrez, A. Cryer, I. Caliskanelli, H. Tugal, R. Skilton (Eds.), Towards Autonomous Robotic Systems, Springer International Publishing, 2022, pp. 114–127, https://doi.org/10.1007/978-3-031-15908-4_10.
[35] W. Luo, B. Yang, R. Urtasun, Fast and Furious: Real Time End-to-End 3D Detection, Tracking and Motion Forecasting with a Single Convolutional Net, IEEE/CVF Conference on Computer Vision and Pattern Recognition 2018 (2018) 3569–3577, https://doi.org/10.1109/CVPR.2018.00376.
[36] L. Ma, Y. Li, J. Li, C. Wang, R. Wang, M.A. Chapman, Mobile laser scanned point-clouds for road object detection and extraction: a review, Remote Sens. (Basel) 10(10), Article 10 (2018), https://doi.org/10.3390/rs10101531.
[37] W.-C. Ma, R. Urtasun, I. Tartavull, I.A. Barsan, S. Wang, M. Bai, G. Mattyus, N. Homayounfar, S.K. Lakshmikanth, A. Pokrovsky, Exploiting sparse semantic HD maps for self-driving vehicle localization, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) 2019 (2019) 5304–5311, https://doi.org/10.1109/IROS40897.2019.8968122.
[38] M.S. Mekala, W. Park, G. Dhiman, G. Srivastava, J.H. Park, H.-Y. Jung, Deep learning inspired object consolidation approaches using LiDAR data for autonomous driving: a review, Arch. Comput. Meth. Eng. (2021), https://doi.org/10.1007/s11831-021-09670-y.
[39] H. Min, K.M. Han, Y.J. Kim, Accelerating probabilistic volumetric mapping using ray-tracing graphics hardware, IEEE Int. Conf. Robot. Autom. (ICRA) 2021 (2021) 5440–5445, https://doi.org/10.1109/ICRA48506.2021.9561068.
[40] B. Nguyen, I. Brilakis, P.A. Vela, Optimized parameters for over-height vehicle detection under variable weather conditions, J. Comput. Civ. Eng. 31 (5) (2017) 04017039, https://doi.org/10.1061/(ASCE)CP.1943-5487.0000685.
[41] U. Okyay, J. Telling, C.L. Glennie, W.E. Dietrich, Airborne lidar change detection: An overview of Earth sciences applications, Earth Sci. Rev. 198 (2019), https://doi.org/10.1016/j.earscirev.2019.102929 102929.
[42] J. Otepka, G. Mandlburger, W. Karel, B. Wöhrer, C. Ressl, N. Pfeifer, A framework for generic spatial search in 3d point clouds, ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, V-2–2021, 2021, 35–42. https://doi.org/10.5194/isprs-annals-V-2-2021-35-2021.
[43] S. Pagad, D. Agarwal, S. Narayanan, K. Rangan, H. Kim, G. Yalla, Robust method for removing dynamic objects from point clouds, IEEE International Conference on Robotics and Automation (ICRA) 2020 (2020) 10765–10771, https://doi.org/10.1109/ICRA40945.2020.9197168.
[44] Y. Pan, B. Gao, J. Mei, S. Geng, C. Li, H. Zhao, SemanticPOSS: a point cloud dataset with large quantity of dynamic instances, IEEE Intelligent Vehicles Symposium (IV) 2020 (2020) 687–693, https://doi.org/10.1109/IV47402.2020.9304596.
[45] A. Petrovskaya, S. Thrun, Model based vehicle detection and tracking for autonomous urban driving, Auton. Robot. 26 (2) (2009) 123–139, https://doi.org/10.1007/s10514-009-9115-1.
[46] N. Pfeifer, J. Falkner, A. Bayr, L. Eysn, C. Ressl, Test charts for evaluating imaging and point cloud quality of mobile mapping systems for urban street space acquisition, Remote Sens. (Basel) 13 (2) (2021) Article 2, https://doi.org/10.3390/rs13020237.
[47] P. Pfreundschuh, H.F.C. Hendrikx, V. Reijgwart, R. Dubé, R. Siegwart, A. Cramariuc, Dynamic Object Aware LiDAR SLAM based on Automatic Generation of Training Data, IEEE International Conference on Robotics and Automation (ICRA) 2021 (2021) 11641–11647, https://doi.org/10.1109/ICRA48506.2021.9560730.
[48] F. Pomerleau, P. Krüsi, F. Colas, P. Furgale, R. Siegwart, Long-term 3D map maintenance in dynamic environments, IEEE International Conference on Robotics and Automation (ICRA) 2014 (2014) 3712–3719, https://doi.org/10.1109/ICRA.2014.6907397.
[49] G. Postica, A. Romanoni, M. Matteucci, Robust moving objects detection in lidar data exploiting visual cues, IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS) 2016 (2016) 1093–1098, https://doi.org/10.1109/IROS.2016.7759185.
[50] J. Schauer, A. Nüchter, The peopleremover—removing dynamic objects from 3-D point cloud data by traversing a voxel occupancy grid, IEEE Rob. Autom. Lett. 3 (3) (2018) 1679–1686, https://doi.org/10.1109/LRA.2018.2801797.
[51] J. Shackleton, B. VanVoorst, J. Hesch, Tracking People with a 360-Degree Lidar, in: 2010 7th IEEE International Conference on Advanced Video and Signal Based Surveillance, 2010, 420–426. https://doi.org/10.1109/AVSS.2010.52.
[52] M. Soilán, B. Riveiro, A. Sánchez-Rodríguez, P. Arias, Safety assessment on pedestrian crossing environments using MLS data, Accid. Anal. Prev. 111 (2018) 328–337, https://doi.org/10.1016/j.aap.2017.12.009.
[53] T.-A. Teo, T.-Y. Shih, Lidar-based change detection and change-type determination in urban areas, Int. J. Remote Sens. 34 (3) (2013) 968–981, https://doi.org/10.1080/01431161.2012.714504.
[54] A.K. Ushani, R.W. Wolcott, J.M. Walls, R.M. Eustice, A learning approach for real-time temporal scene flow estimation from LIDAR data, IEEE Int. Conf. Robot. Autom. (ICRA) 2017 (2017) 5666–5673, https://doi.org/10.1109/ICRA.2017.7989666.
[55] V. Ussyshkin, L. Theriault, Airborne Lidar: Advances in Discrete Return Technology for 3D Vegetation Mapping, Remote Sens. (Basel) 3 (3) (2011) 416–434, https://doi.org/10.3390/rs3030416.
[56] M. Weinmann, B. Jutzi, S. Hinz, C. Mallet, Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers, ISPRS J. Photogramm. Remote Sens. 105 (2015) 286–304, https://doi.org/10.1016/j.isprsjprs.2015.01.016.
[57] W.W. Wen, G. Zhang, L.-T. Hsu, GNSS NLOS Exclusion Based on dynamic object detection using LiDAR Point Cloud, IEEE Trans. Intell. Transp. Syst. 22 (2) (2021) 853–862, https://doi.org/10.1109/TITS.2019.2961128.
[58] Y. Wu, Y. Wang, S. Zhang, H. Ogai, Deep 3D object detection networks using LiDAR data: a review, IEEE Sens. J. 21 (2) (2021) 1152–1171, https://doi.org/10.1109/JSEN.2020.3020626.
[59] Y. Xia, Z. Sun, A. Tok, S. Ritchie, A dense background representation method for traffic surveillance based on roadside LiDAR, Opt. Lasers Eng. 152 (2022), https://doi.org/10.1016/j.optlaseng.2022.106982 106982.
[60] S. Xu, P. Cheng, Y. Zhang, P. Ding, Error analysis and accuracy assessment of mobile laser scanning system, Open Autom. Control Syst. J. 7 (1) (2015) 485–495, https://doi.org/10.2174/1874444301507010485.
[61] S. Xu, S. Oude Elberink, G. Vosselman, Entities and features for classifcation of airborne laser scanning data in urban area, ISPRS Annals Photogram., Remote Sens. Spatial Inf. Sci. I–4 (2012) 257–262, https://doi.org/10.5194/isprsannals-I-4-257-2012.
[62] Y. Xu, U. Stilla, Toward building and civil infrastructure reconstruction from point clouds: a review on data and key techniques, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 14 (2021) 2857–2885, https://doi.org/10.1109/JSTARS.2021.3060568.
[63] B. Yang, R. Guo, M. Liang, S. Casas, R. Urtasun, RadarNet: exploiting radar for robust perception of dynamic objects, in: A. Vedaldi, H. Bischof, T. Brox, & J.-M. Frahm (Eds.), Computer

Vision – ECCV 2020, 2020, pp. 496–512. Springer International Publishing. https://doi.org/10.1007/978-3-030-58523-5_29.
[64] H. Yin, Y. Wang, X. Ding, L. Tang, S. Huang, R. Xiong, 3D LiDAR-based global localization using siamese neural network, IEEE Trans. Intell. Transp. Syst. 21 (4) (2020) 1380–1392, https://doi.org/10.1109/TITS.2019.2905046.
[65] D. Yoon, T. Tang, T. Barfoot, Mapless online detection of dynamic objects in 3D Lidar, in: 2019 16th Conference on Computer and Robot Vision (CRV), 2019, 113–120. https://doi.org/10.1109/CRV.2019.00023.
[66] L. Zhang, Q. Li, M. Li, Q. Mao, A. Nüchter, Multiple vehicle-like target tracking based on the Velodyne LiDAR*, IFAC Proceedings Volumes 46 (10) (2013) 126–131, https://doi.org/10.3182/20130626-3-AU-2035.00058.
[67] T. Zhang, Y. Nakamura, Moving Humans Removal for Dynamic Environment Reconstruction from Slow-Scanning LIDAR Data, in: 2018 15th International Conference on Ubiquitous Robots (UR), 2018, 449–454. https://doi.org/10.1109/URAI.2018.8441778.