Master Thesis
Dynamic Objects Detection and Removal in Mobile Laser Scanning Data
P5 Presentation

First Supervisor
Peter van Oosterom (TUDelft)
Second Supervisor
Jesús Balado Frías (TUDelft&UVigo)

Industry mentors
Bart Beers (Cyclomedia technology)
Arjen Swart (Cyclomedia technology)

Delegate of the Board of Examiners
Dirk Dubbeling (TUDelft)
Co-reader
Mieke Kuschnerus (TUDelft)

Author
Zhenyu Liu (5386586)

27 June 2022

Introduction

Mobile Laser Scanning (**MLS**) provides an efficient way to capture 3D spatial data with high geometric accuracy and rich details from the real world. These advantages make MLS fit the demand for point clouds data in **urban scenes**, especially in **linear road environments**. So in recent years, MLS has been widely used in many urban applications such as **urban land cover analysis**, **urban environment monitoring**, **digital 3D modeling**, and **self-driving vehicles**.

Introduction

- **Introduction**
- Targets
- Related Work
- Methodology
- Implementation
- Conclusion
- Future Work

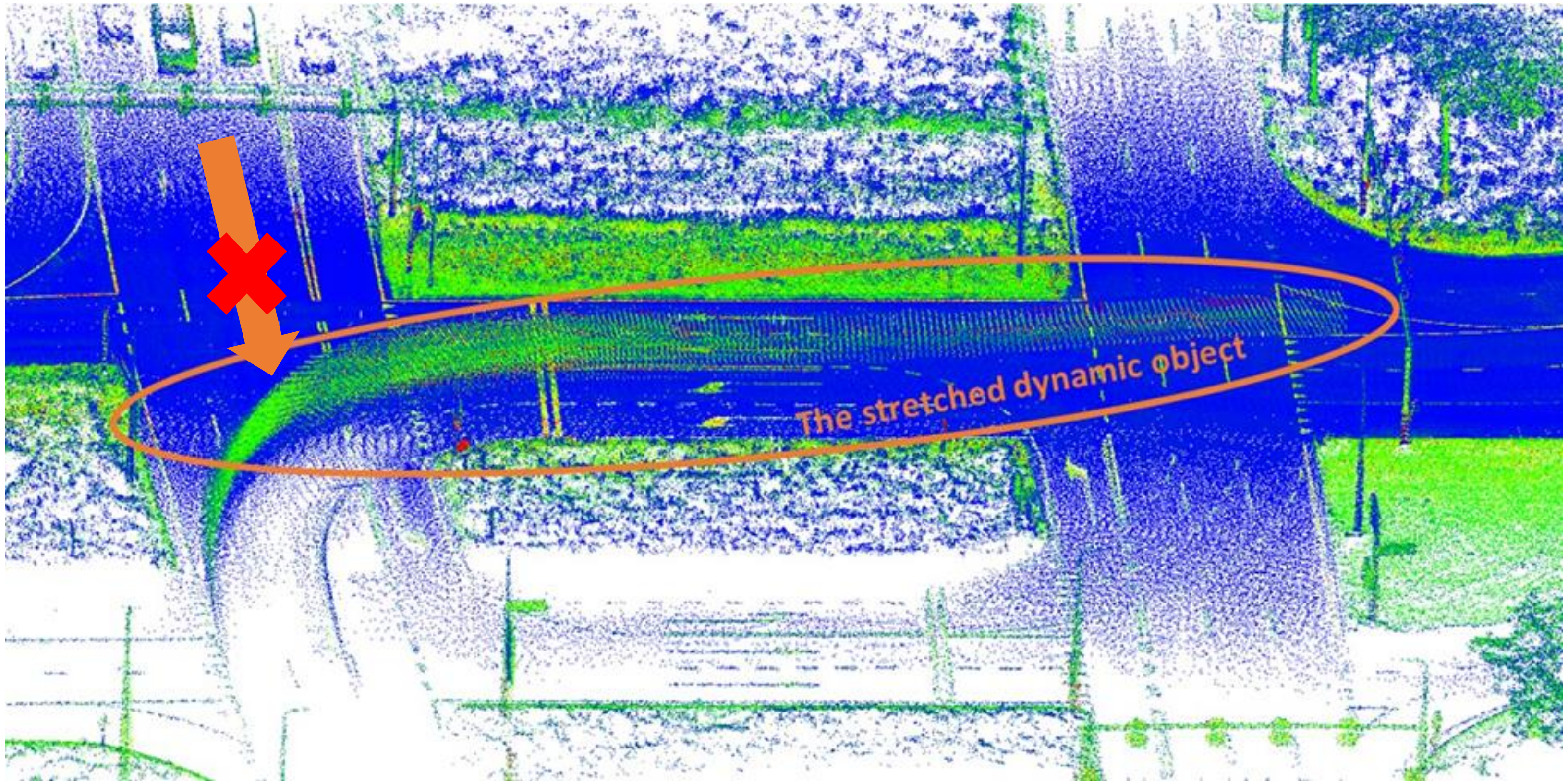
MLS systems are usually mounted on land-based mobile platforms such as vehicles

Introduction

Due to the data acquisition method of MLS, it is impossible to completely avoid dynamic objects in the original point cloud data. The dynamic object is one the main problems in MLS data:

- Many application scenarios **only need static environments**. Dynamic objects can affect the performance of these applications, such as navigation and localization.
- MLS data has more obvious **ghost trail effect** than other types of point cloud data.

Introduction



The stretched dynamic object/ghost trail effect in MLS data

The stretched dynamic object forms a wall in the middle of the road making the navigation algorithm consider this road as impassable.

Targets

The main research question of this thesis is as follows:

"How to detect and remove the dynamic objects from the MLS data?"

After defining the main question of this research, some sub-questions are derived from it:

- **How to detect and remove dynamic objects and avoid residue?**
- **How to avoid detecting and removing static environment objects?**
- **What factors affect the detection results?**
- **How to use MLS sensor trajectory to assist detection and removal operations?**
- **What types of objects often lead to misdetection?**
- **How to improve the computational efficiency for large-scale data?**

Related Work

Single-frame Data Methods VS Multi-frame Data Methods

The single-frame lidar data refers to the scan data obtained by the sensor in an instant or 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.

Related Work

Single-frame Data Methods VS Multi-frame Data Methods

The single-frame lidar data

- **Prior Map (background subtraction)** → **Difficult to create high quality prior map**
- **Feature or model matching** → **Need prior knowledge/can't handle unknown objects**
- **Motion Artifacts** → **The trajectory and speed of MLS dynamic objects are variable**
- **Deep Learning** → **Need high quality training set/training samples are very imbalanced**

Overall, it is still challenging to extract dynamic objects in MLS data using single-frame data approaches.

Related Work

Single-frame Data Methods VS Multi-frame Data Methods

Multi-frame data is a collection of single-frame data, such as the continuously scanning MLS data is a typical multi-frame data. The spatiotemporal relations between different data frames provide much useful information especially for detection of dynamic objects.

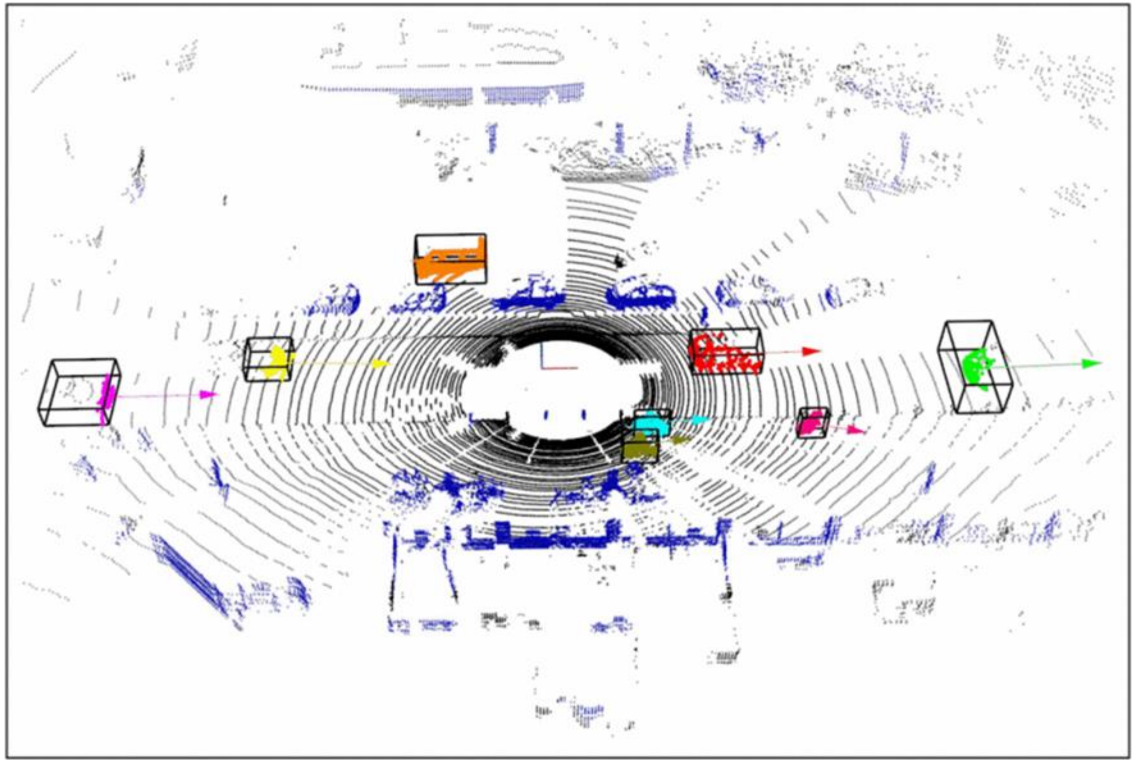
Related Work

Single-frame Data Methods VS Multi-frame Data Methods

Multi-frame data

•Object Tracking

→ Need prior knowledge/can't handle unknown objects/poor generalization ability



Objects segmented with motion cues (Dewan et al., 2016).

Related Work

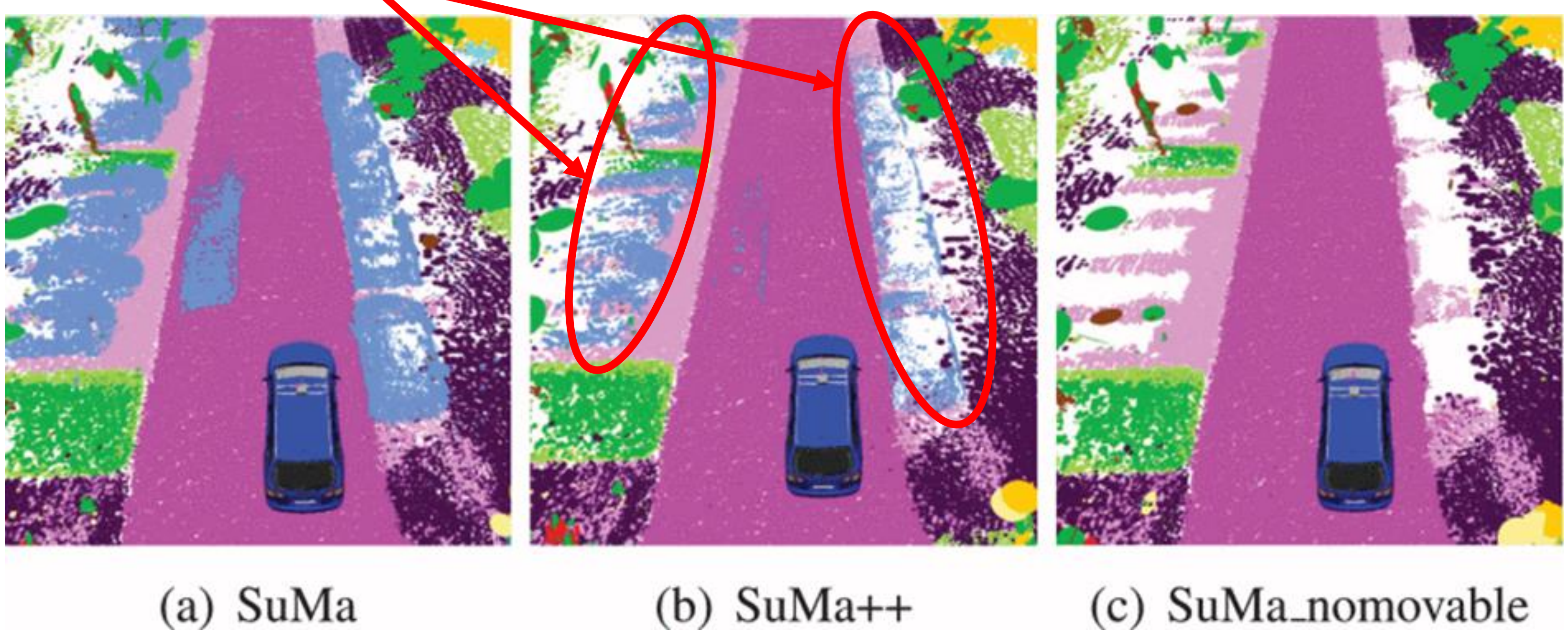
Single-frame Data Methods VS Multi-frame Data Methods

Multi-frame data

•Point-based Dynamic Detection

→ Does not consider the noise and objects with sparse structure

→ Incomplete removal



The dynamic object removal in SuMa++ (X. Chen et al., 2019).

Related Work

Single-frame Data Methods VS Multi-frame Data Methods

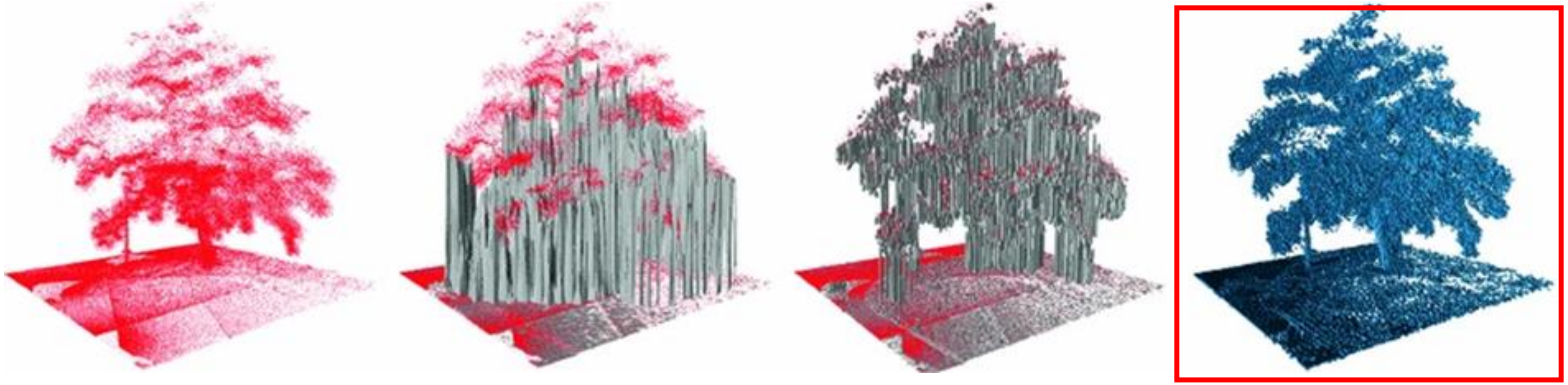
Multi-frame data

•Octomap

→ Ray-based method

→ Instead of simply marking points as static or dynamic, Octomap describes the probability of each voxel being occupied, which considers the effect of noise base on ray tracing.

→ Not compute-friendly and memory-friendly

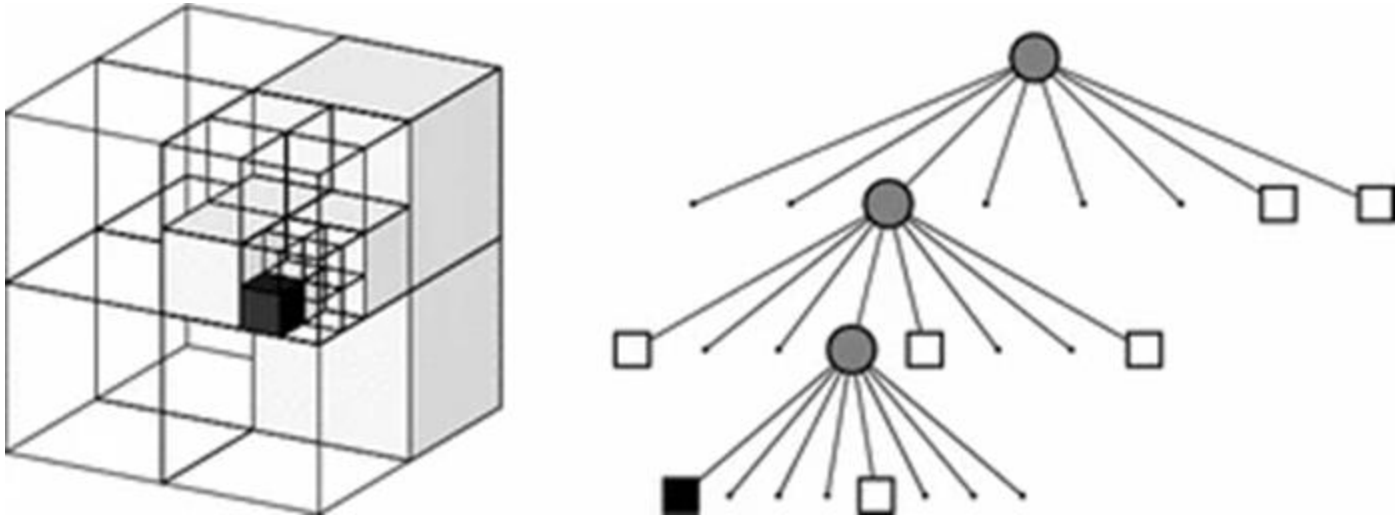


3D representations of a tree scanned with a laser range sensor (from left to right): Point cloud, elevation map, multi-level surface map, and **Octomap** (Hornung et al., 2013).

Related Work

Octomap

Octomap is a 3D Voxel-based Mapping Framework based on Octrees.

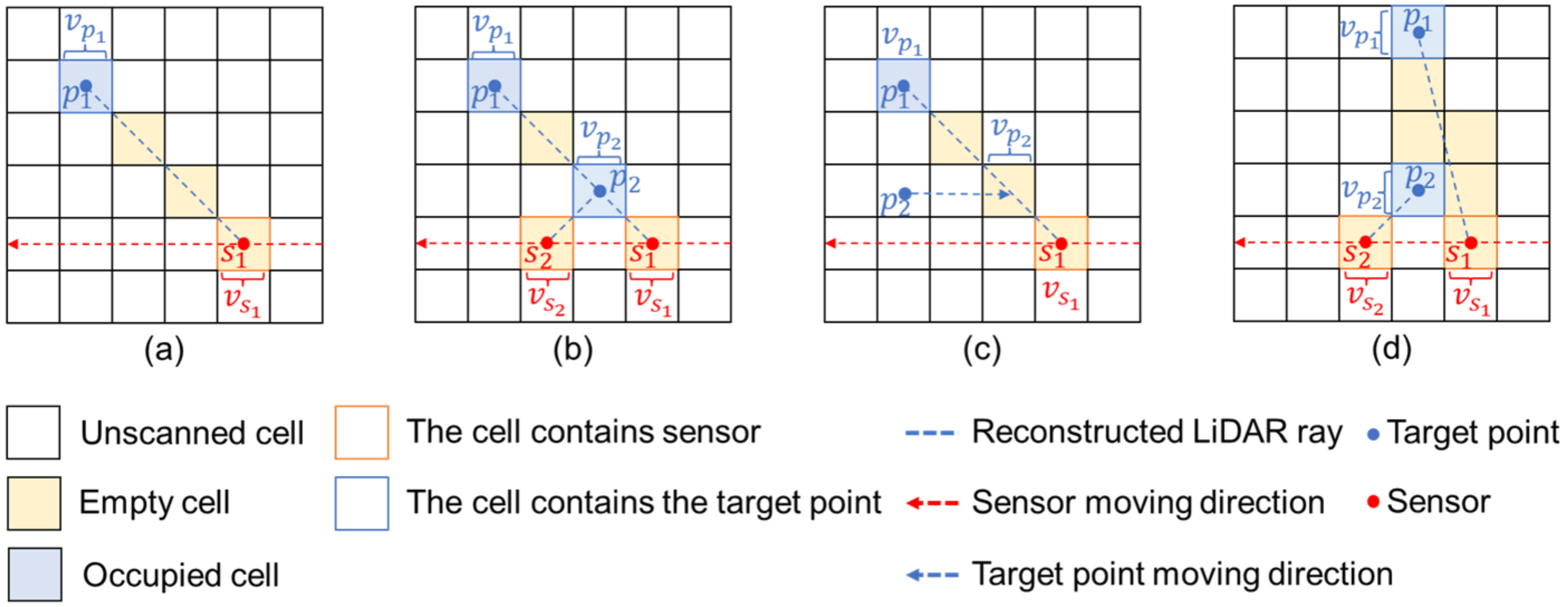


The volumetric model (left) and its corresponding octree representation (right) (Hornung et al., 2013).

Related Work

Octomap

Octomap can use spatial conflicts between LiDAR rays to estimate the probability that a space is occupied.

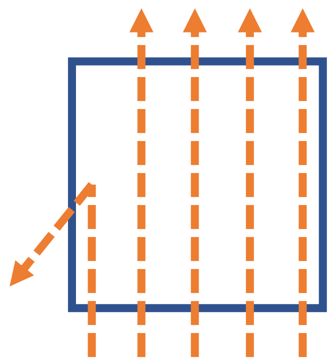


Related Work

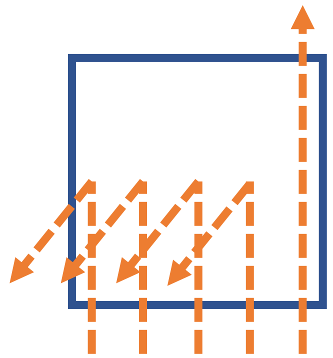
Octomap

Octomap uses the occupancy probability (OP) to estimate if a voxel cell is occupied by static points.

$$OP = \frac{Number\ of\ rays\ reflected\ from\ the\ cell}{Number\ of\ rays\ reaching\ the\ cell} \cdot 100\%$$



$$OP1 = \frac{1}{5} \cdot 100\% = 20\%$$

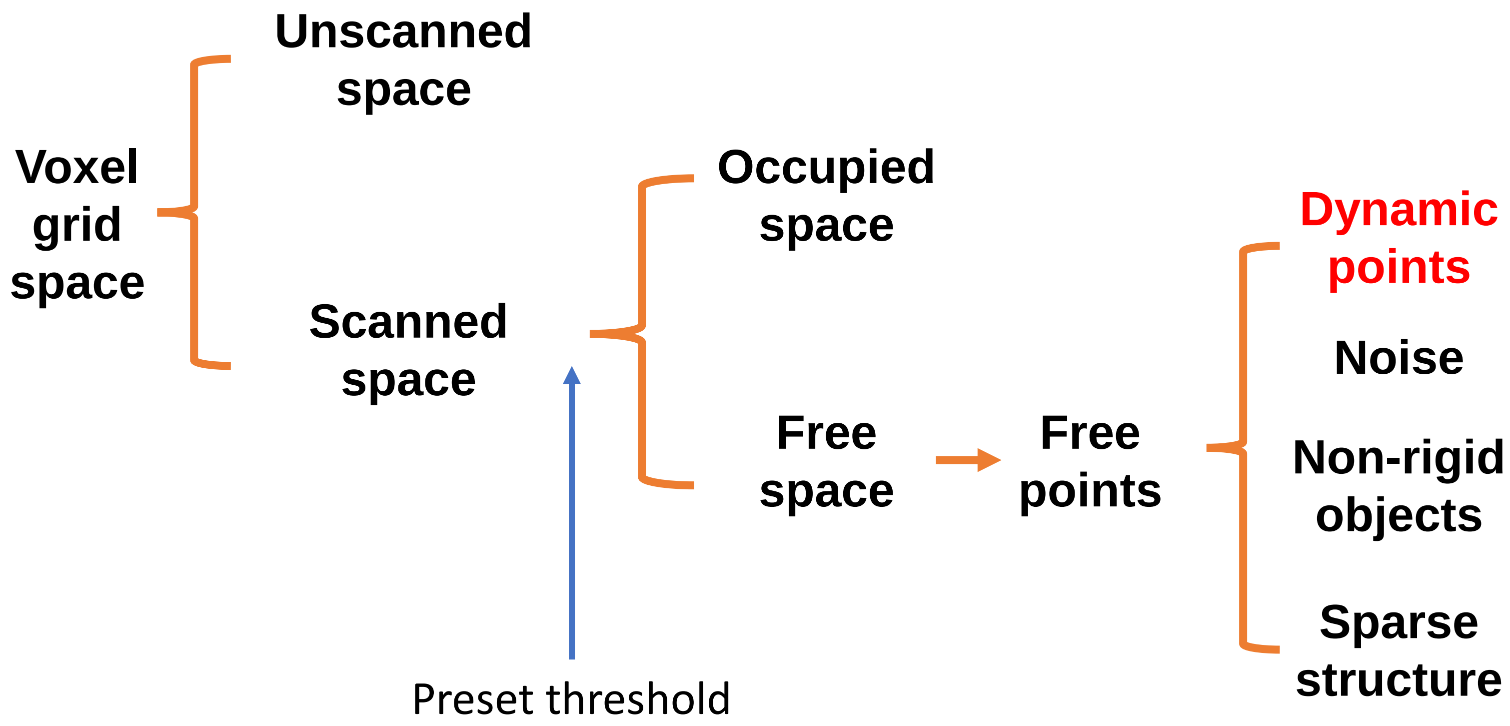


$$OP2 = \frac{4}{5} \cdot 100\% = 80\%$$

Related Work

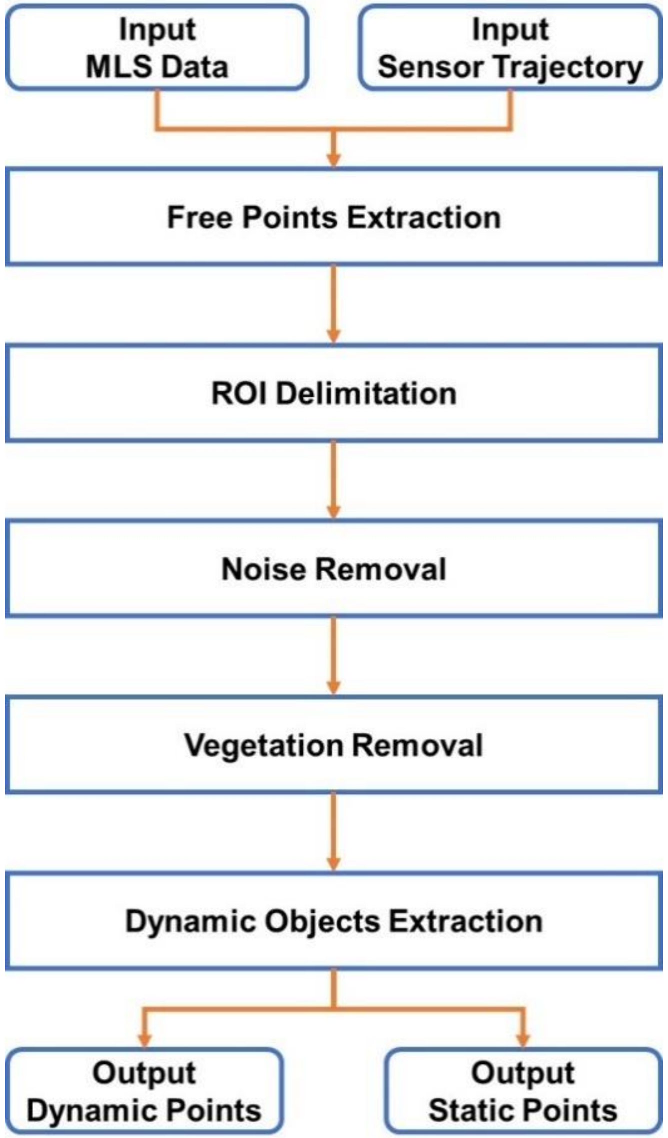
Octomap

Octomap uses the occupancy probability (OP) to estimate if a voxel cell is occupied by static points.



- Introduction
- Targets
- **Related Work**
- Methodology
- Implementation
- Conclusion
- Future Work

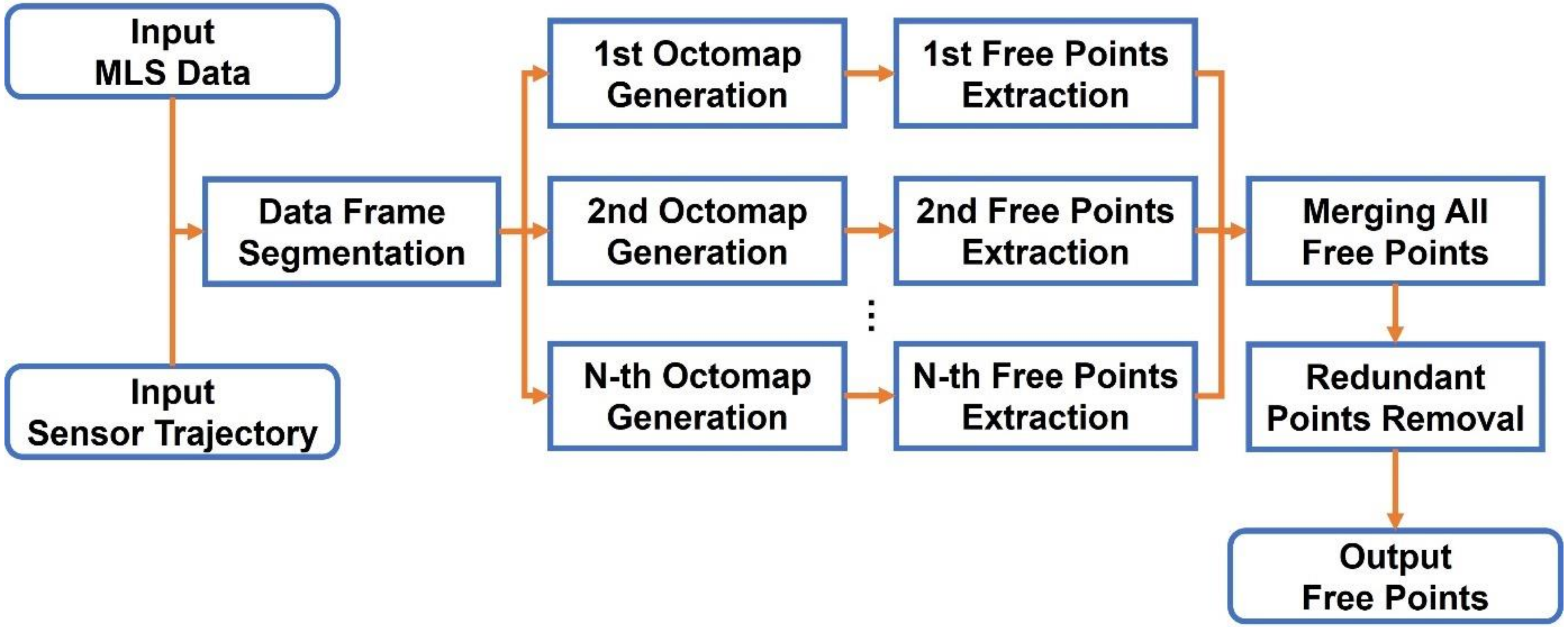
Methodology



The main workflow of proposed method

Methodology

Free Point Extraction

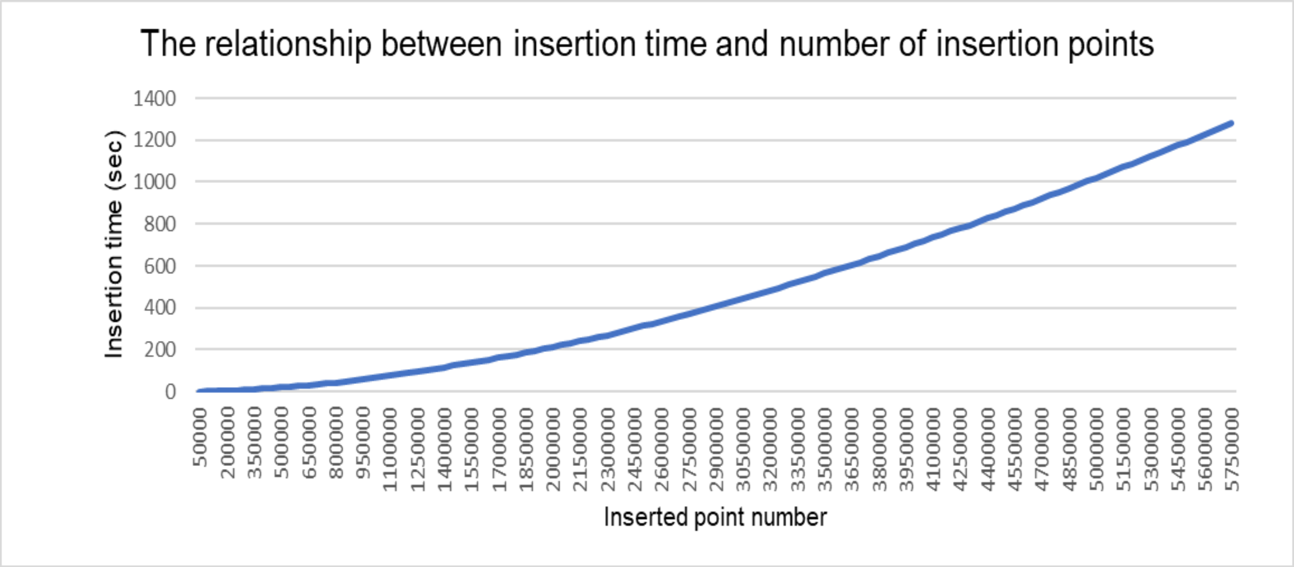


The workflow of free point extraction

Methodology

Free Point Extraction

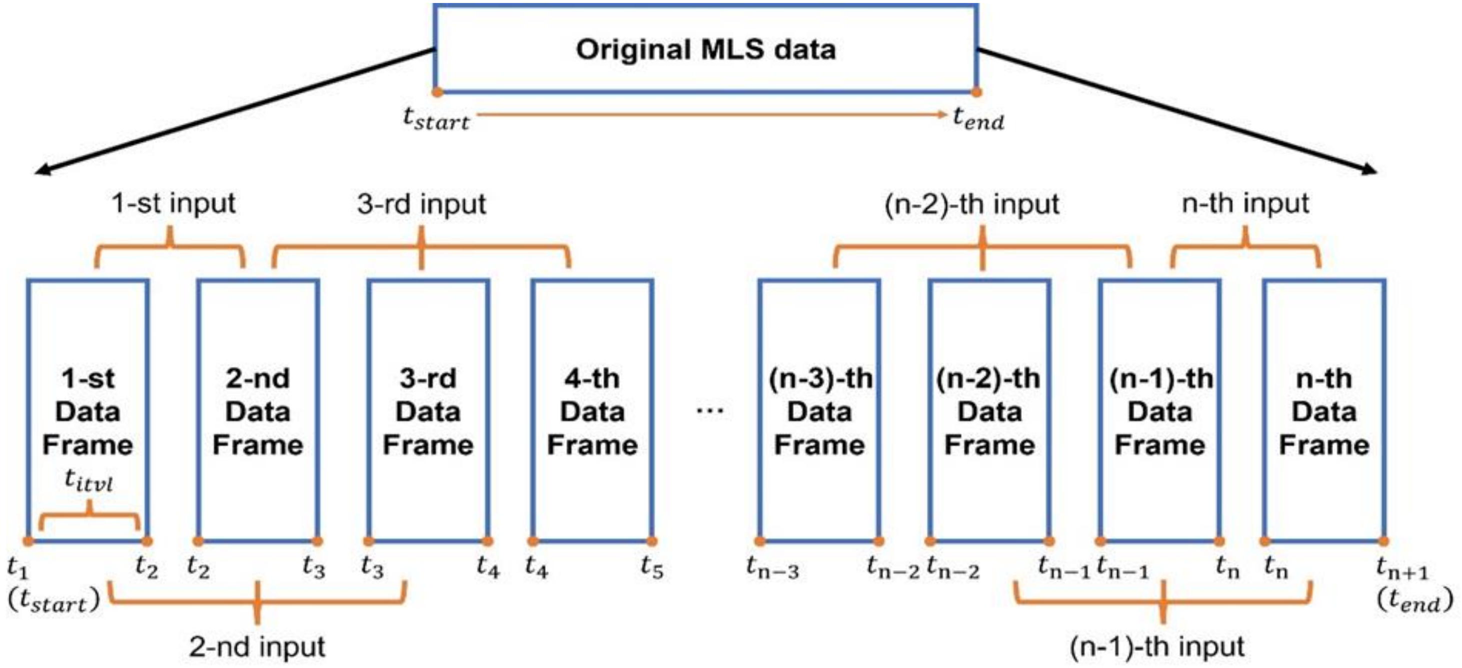
Octomap needs to be improved in terms of computational efficiency and memory consumption. This proposed method segment the original input MLS point cloud into multiple subsets base on GPS time thus avoiding the generation of a huge voxel grid to achieve better efficiency and reduced memory requirements.



Methodology

Free Point Extraction

Octomap needs to be improved in terms of computational efficiency and memory consumption. This proposed method segment the original input MLS point cloud into multiple subsets base on GPS time thus avoiding the generation of a huge voxel grid to achieve better efficiency and reduced memory requirements.

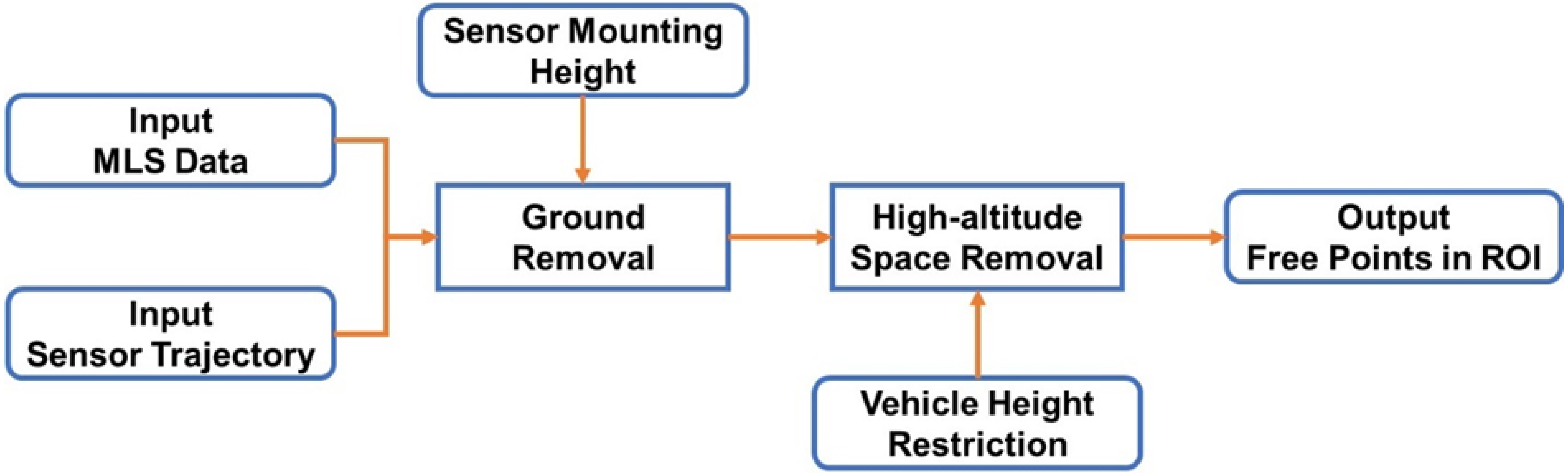


Data frames segmentation

Methodology

ROI Delimitation

Using the sensor trajectory, sensor mounting height, and local vehicle height restriction information to extract Region of Interest (ROI) that is relevant for land-based dynamic objects. ROI is defined as the space between the height of the ground surface and the maximum allowable height of a large vehicle in this research (excluding the ground).



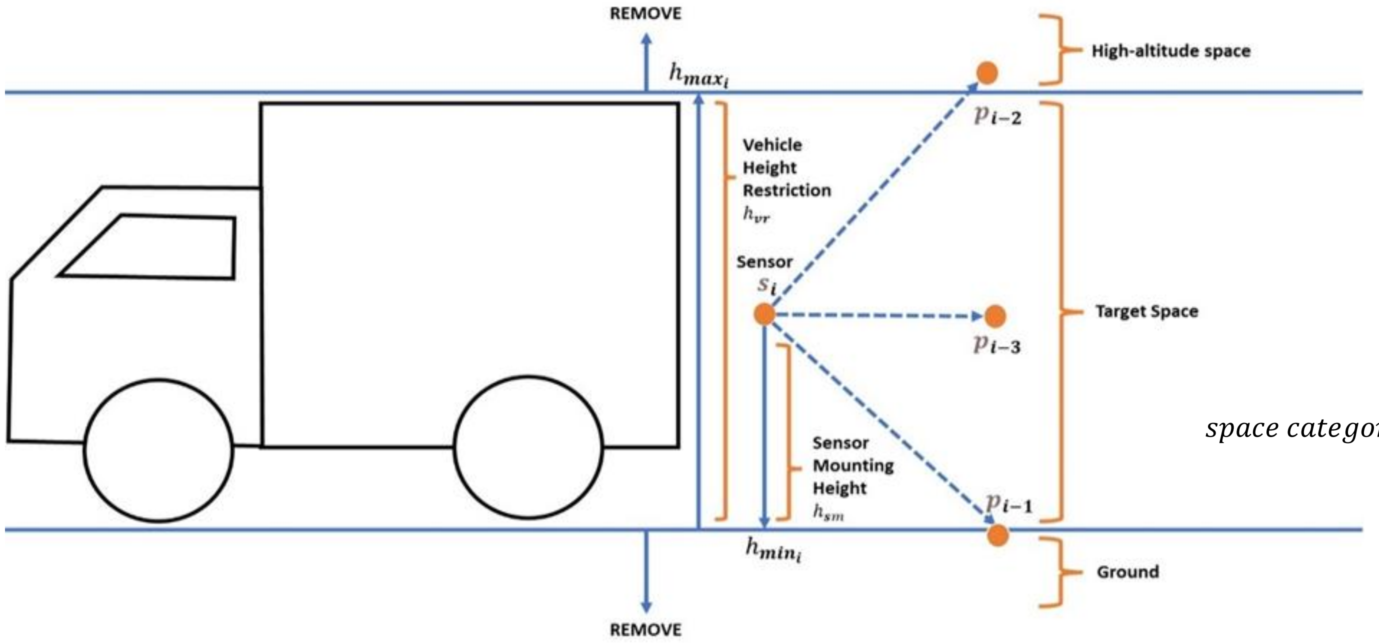
The workflow of ROI delimitation

Methodology

- Introduction
- Targets
- Related Work
- **Methodology**
- Implementation
- Conclusion
- Future Work

ROI Delimitation

Using the sensor trajectory, sensor mounting height, and local vehicle height restriction information to extract Region of Interest (ROI) that is relevant for land-based dynamic objects. ROI is defined as the space between the height of the ground surface and the maximum allowable height of a large vehicle in this research (excluding the ground).

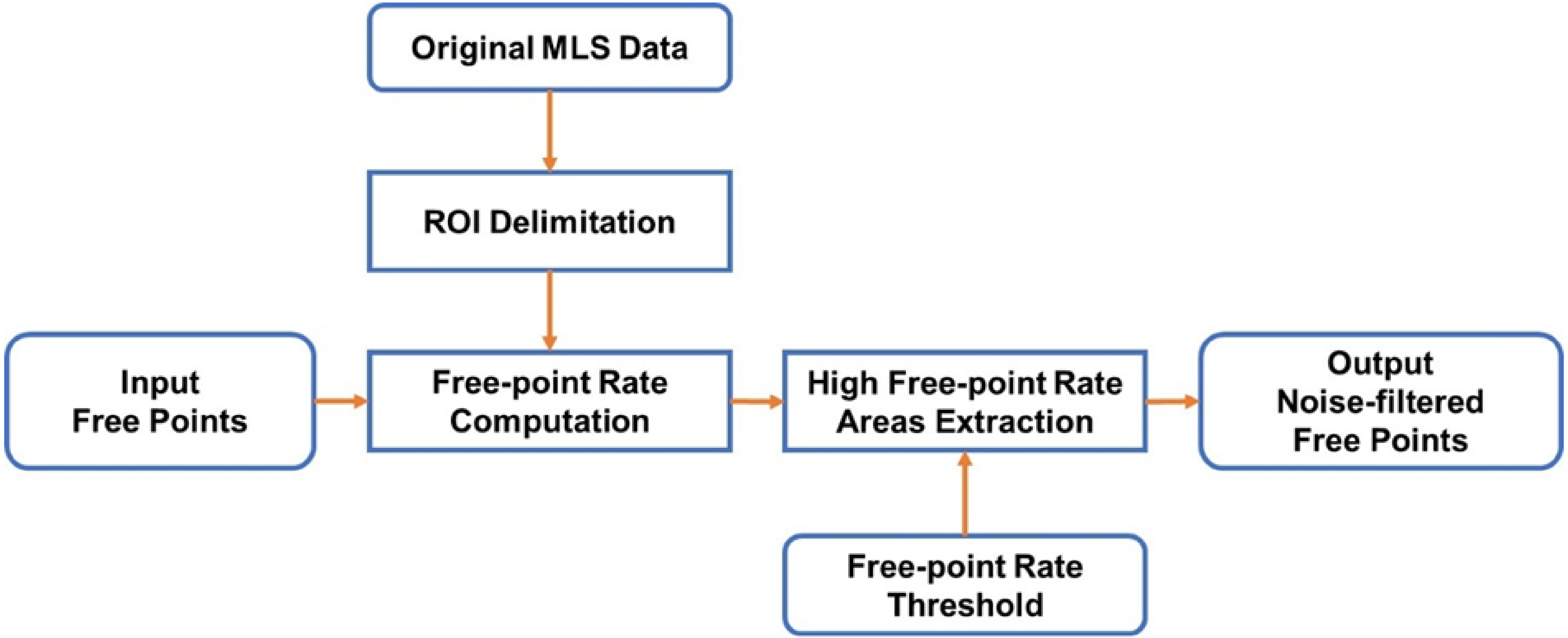


$$space\ category = \begin{cases} ground: if\ h_{min_i} \geq h_{p_i} \\ target\ space: h_{min_i} < h_{p_i} < h_{max_i} \\ high-altitude\ space: if\ h_{max_i} \leq h_{p_i} \end{cases}$$

Spatial segmentation discriminant model

Methodology

Noise Removal



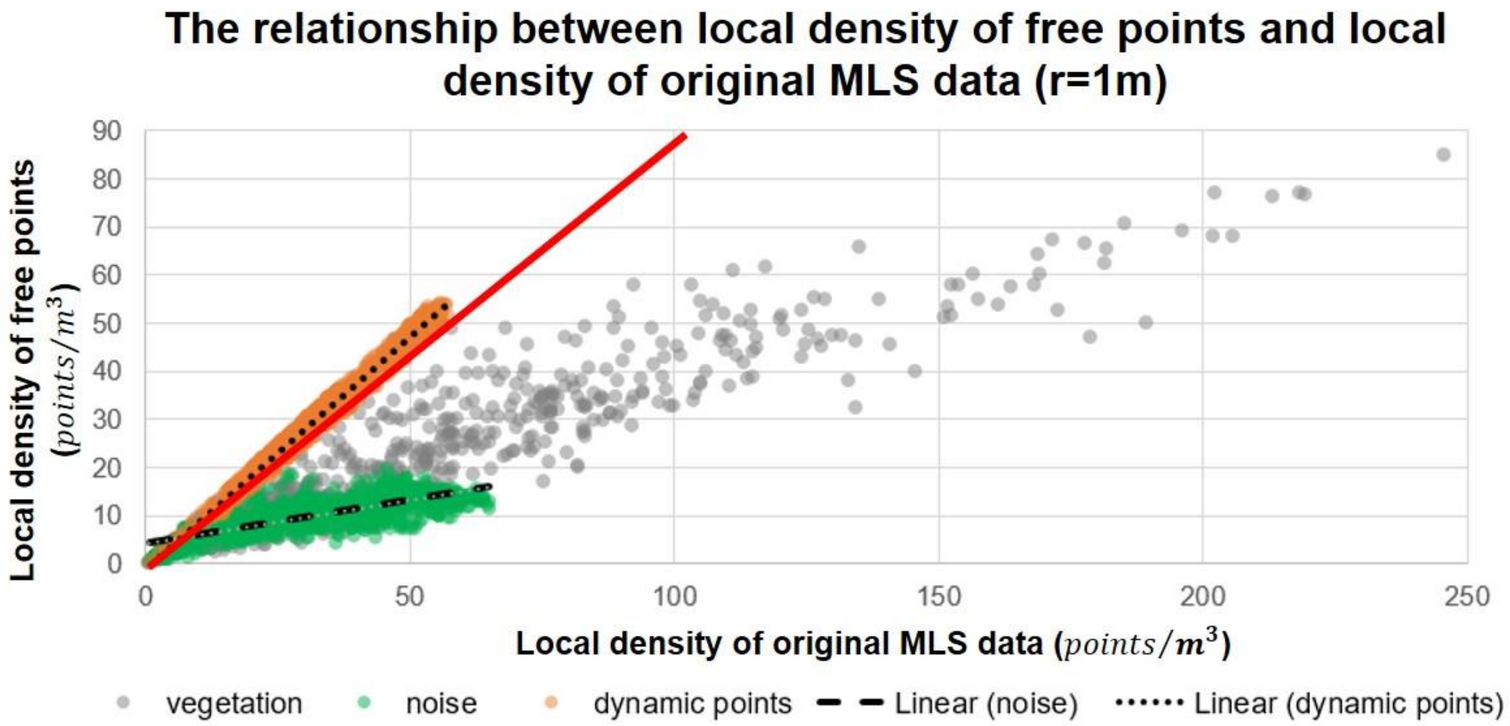
The workflow of noise removal

Methodology

Noise Removal

Noise caused by measurement errors is removed from the free points based on the free-point rate.

For dynamic point, most of its neighbors are also dynamic points.
But for noise, only a few of its neighbors are also noise.



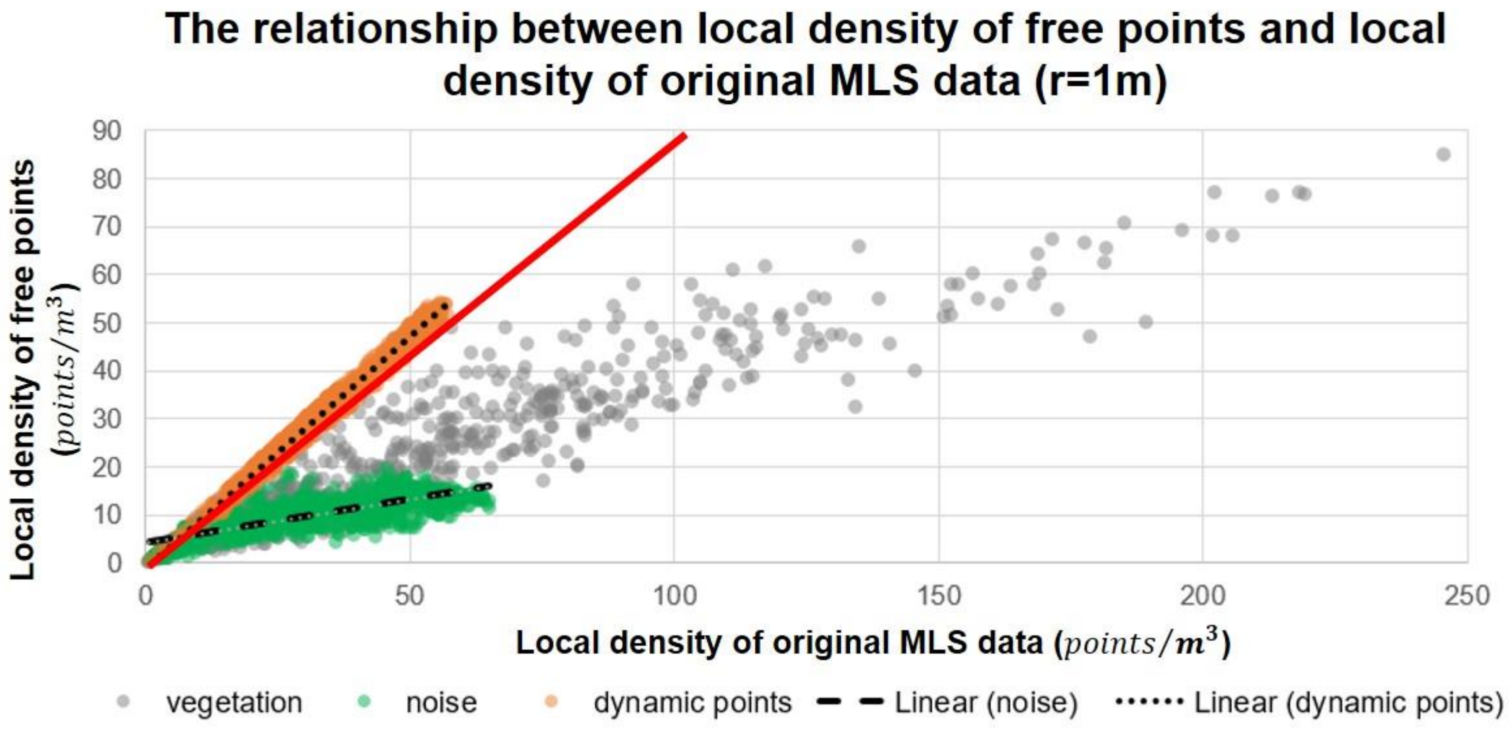
The relationship between the local density of free points and the local density of original MLS data (r=1m).

Methodology

Noise Removal

Noise caused by measurement errors is removed from the free points based on the free-point rate.

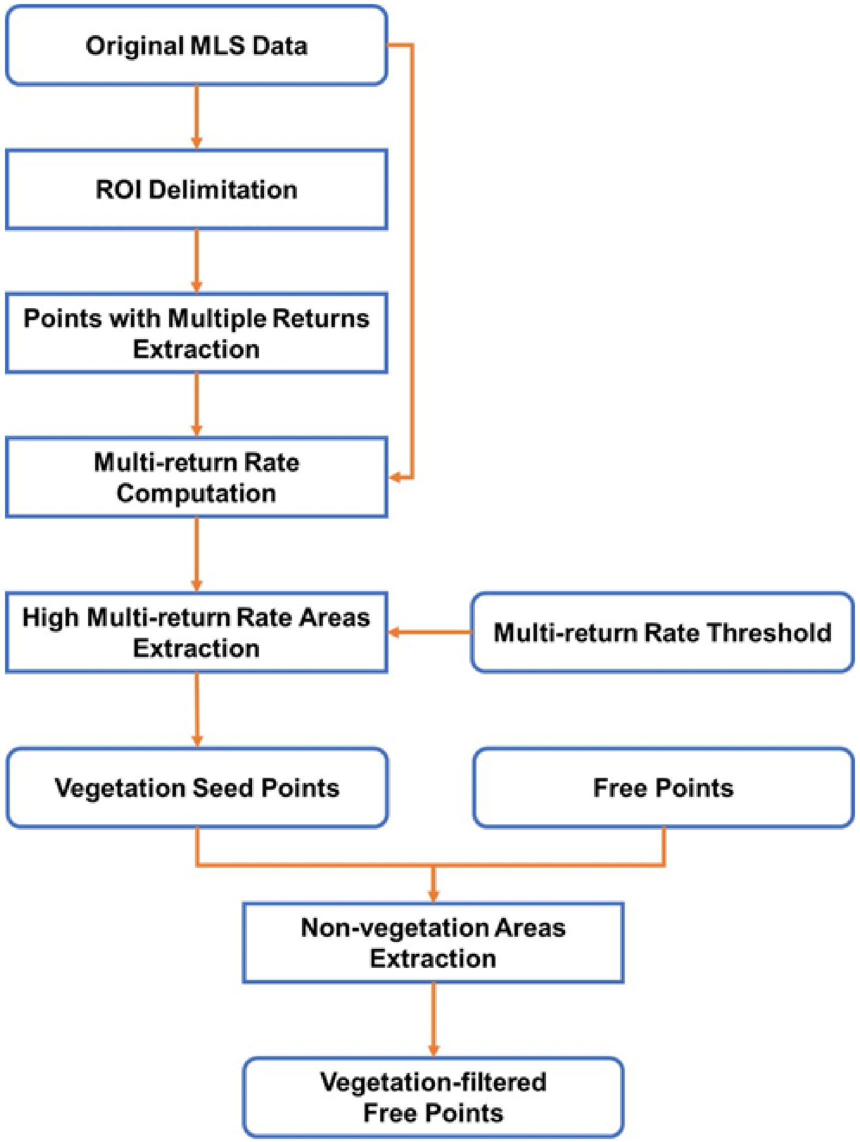
$$free - point\ rate = \frac{Number\ of\ free\ pts\ in\ neighborhood}{Number\ of\ all\ scanned\ pts\ in\ neighborhood} \cdot 100\%$$



The relationship between the local density of free points and the local density of original MLS data (r=1m).

Methodology

Vegetation Removal



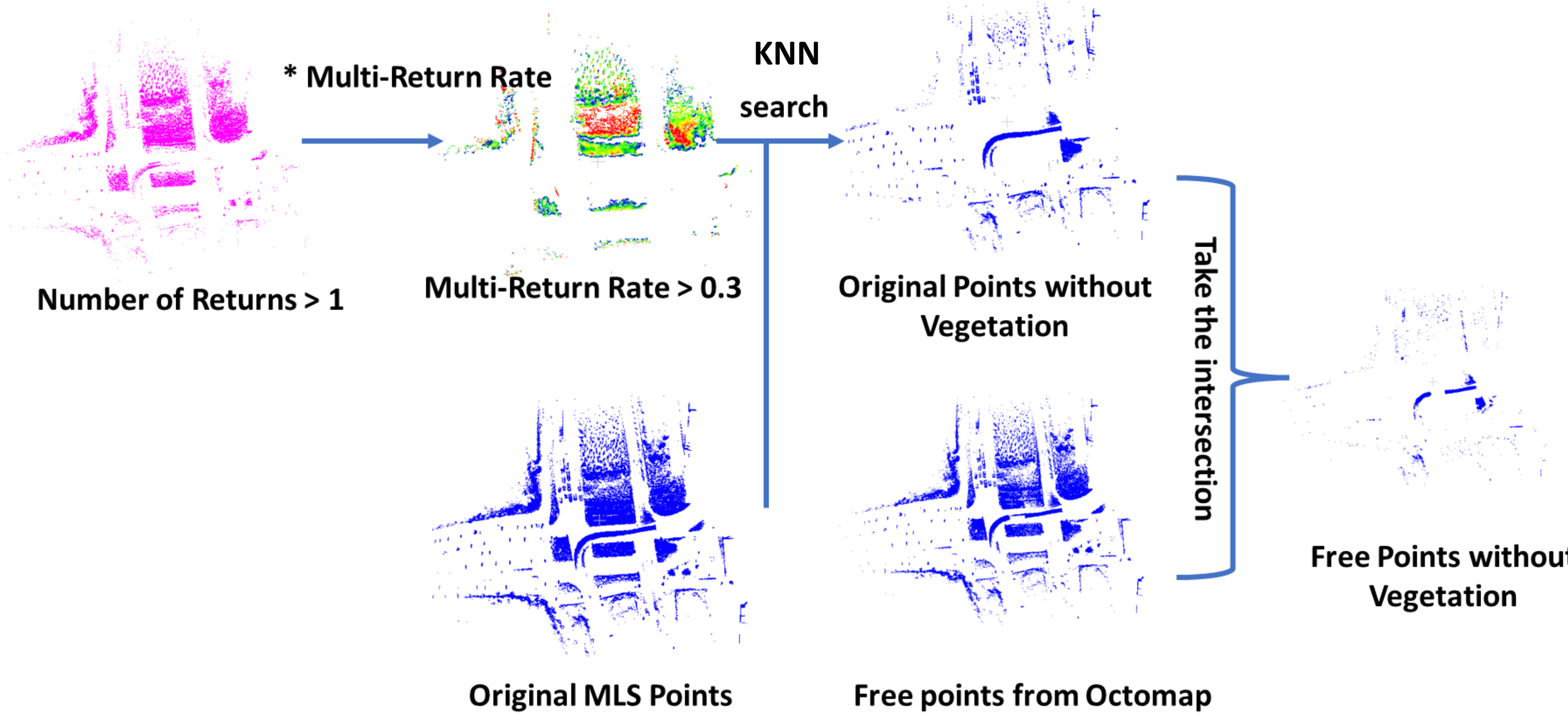
The workflow of vegetation removal.

Methodology

Vegetation Removal

Vegetation is removed from free points using the number of returned LiDAR rays.

$$multi - return\ rate = \frac{Number\ of\ pts\ with\ multiple\ returns\ in\ neighborhood}{Number\ of\ all\ scanned\ pts\ in\ neighborhood} \cdot 100\%$$



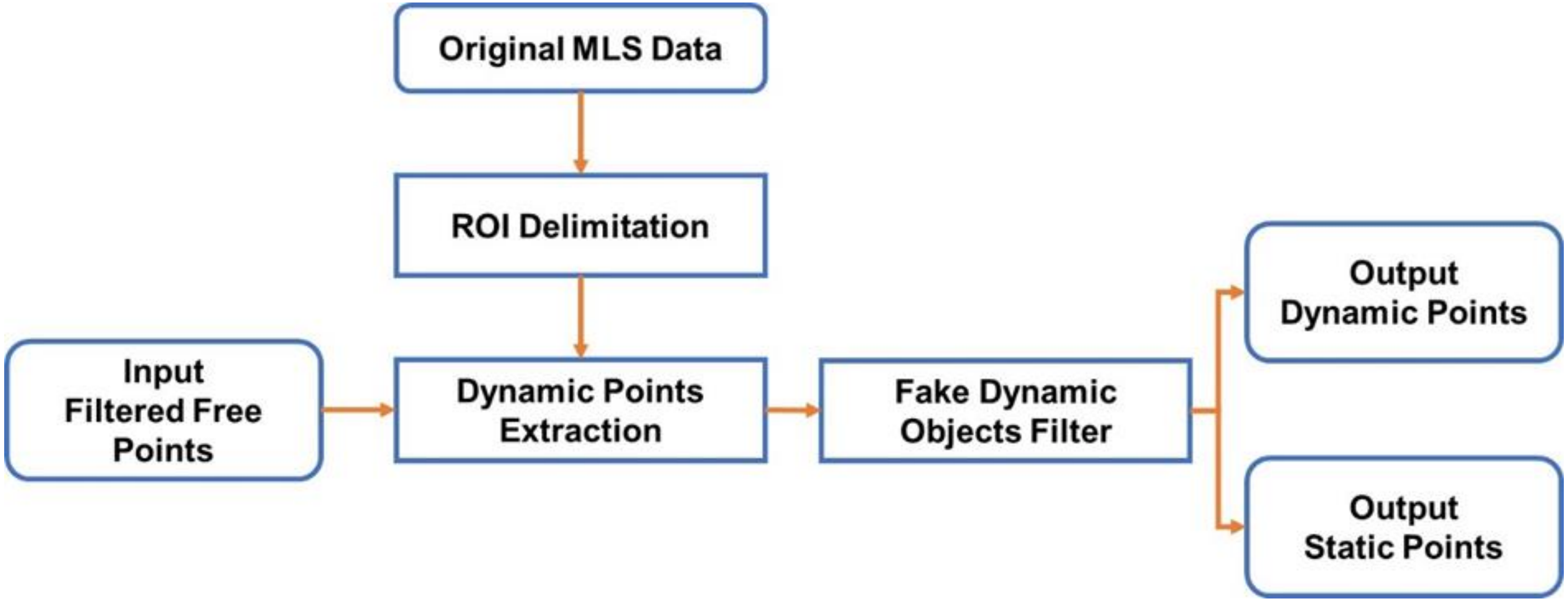
The density of the original point is higher than that of the free point, so the effect of KNN search is better than that of the free point.

The workflow of vegetation removal.

Methodology

Dynamic Objects Extraction

Extracting dynamic objects from the original MLS point cloud using filtered free points as seed points using KNN spatial search.



The workflow of dynamic objects extraction.

Methodology

Dynamic Objects Extraction

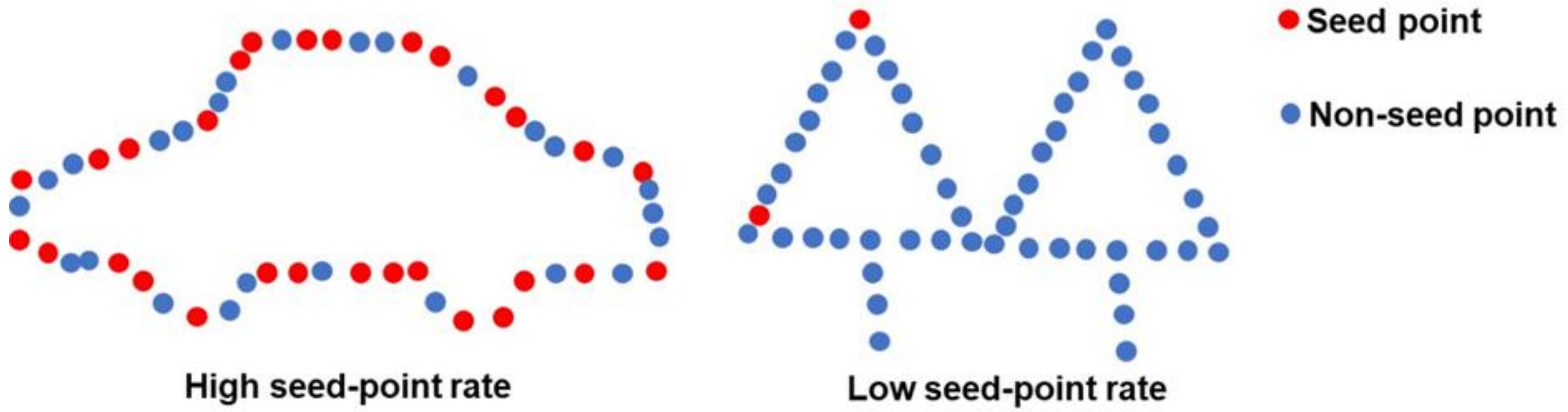
Filtering fake dynamic objects:

- **returning an object that contains only a few points.**

Set a minimum point limit to remove objects with too few points.

- **returning an object with a small proportion of seed points.**

$$seed\text{—}point\ rate = \frac{Number\ of\ seed\ pts\ in\ neighborhood}{Number\ of\ all\ scanned\ pts\ in\ neighborhood} \cdot 100\%$$



The workflow of dynamic objects extraction.

Implementation

Data

Case Site	Full Name of the Case Site	Scan Time (sec)	Number of Points
Position A	The junction of Voorhofdreef, Tanthofdreef, and Kruithuisweg	10	3637969
Position B	The junction of Voorhofdreef, Menno Ter Braaklaan, and Bosboom-Toussaintplein	10	4616356
Position C	The junction of Voorhofdreef, J.J. Slauerhofflaan, and Frederik van Eedenlaan	10	4666430
Position D	The junction of Tanthofdreef and Forensenweg	10	4684840

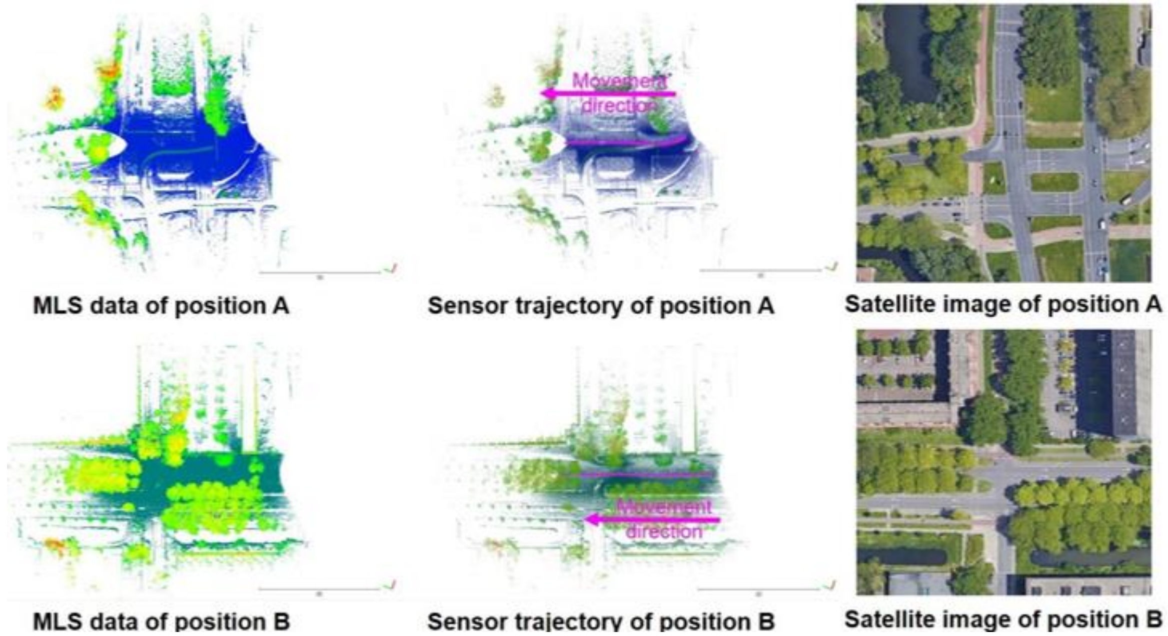
Four case sites.

*The data is provided by Cyclomedia.

Implementation

Data

These four case sites are all located at road junctions and have bicycle lanes, so it is easier to find different types of dynamic objects, such as vehicles and bicycles. Dynamic objects with different speeds and moving directions can be observed in these areas.

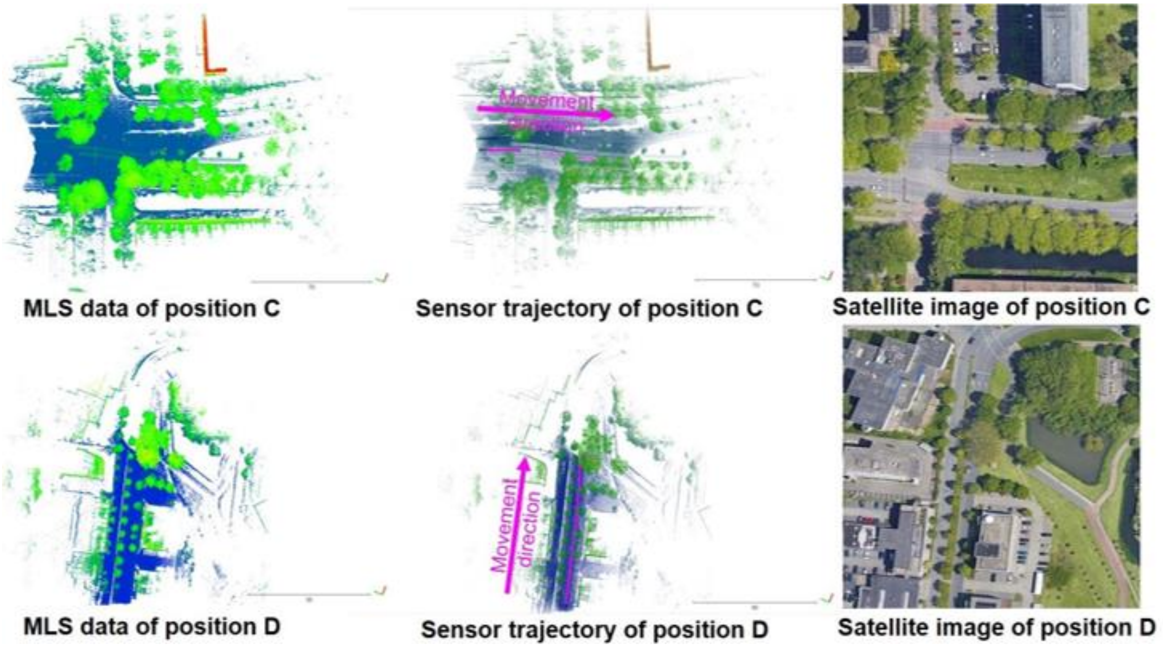


MLS data (left column) rendered in height, corresponding sensor trajectories (middle column) colorized in the purple, and satellite images (right column) from Google Map of the four case sites.

Implementation

Data

These four case sites are all located at road junctions and have bicycle lanes, so it is easier to find different types of dynamic objects, such as vehicles and bicycles. Dynamic objects with different speeds and moving directions can be observed in these areas.



MLS data (left column) rendered in height, corresponding sensor trajectories (middle column) colorized in the purple, and satellite images (right column) from Google Map of the four case sites.

Implementation

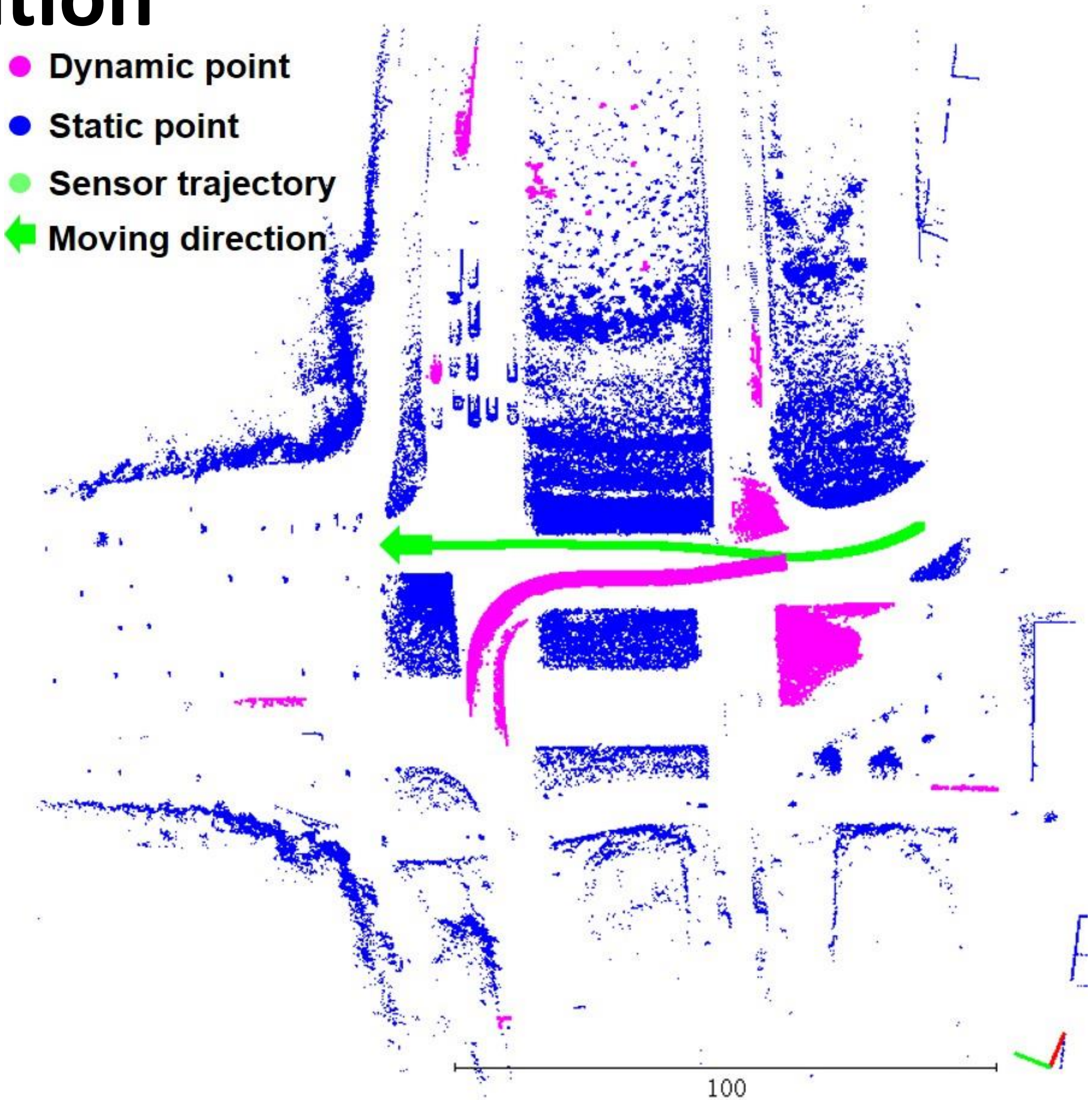
Parameters

Full Name of the Parameter	Parameter	Value
Time interval for data frame segmentation	t_{itvl}	0.75 sec
Octomap voxel size	$size_{voxel}$	0.2 m
Occupancy probability threshold	$thres_{occupied}$	0.7
Sensor mounting height	h_{sm}	2 m
Height restriction of large vehicles	h_{vr}	4 m
Free-point rate threshold	$thres_{fp}$	0.9
Neighborhood radius used to calculate the free-point rate	r_{ns}	1 m
Multi-return rate threshold	$thres_{mr}$	0.3
Neighborhood radius used to calculate the multi-return rate	r_{vg}	1 m
Number of nearest neighbors used to extract all vegetation points	k_{vg}	5
Number of nearest neighbors used to extract all dynamic points	k_{do}.	5
Minimum point number limit for dynamic objects	num_{min}	15
Threshold of seed point proportion	$thres_{sp}$	0.03

Values of implementation parameters

Implementation

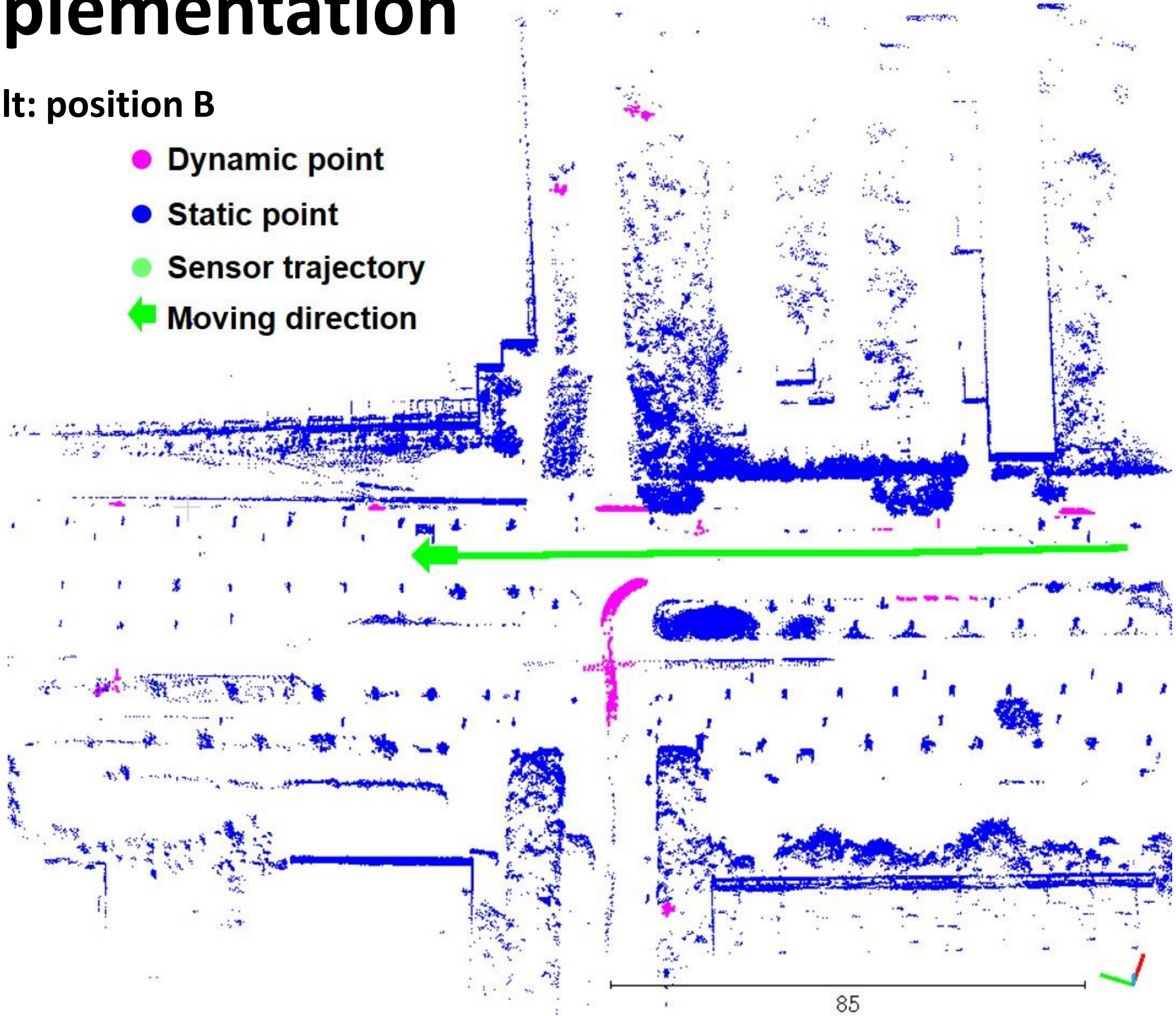
Result: position A



- Introduction
- Targets
- Related Work
- Methodology
- **Implementation**
- Conclusion
- Future Work

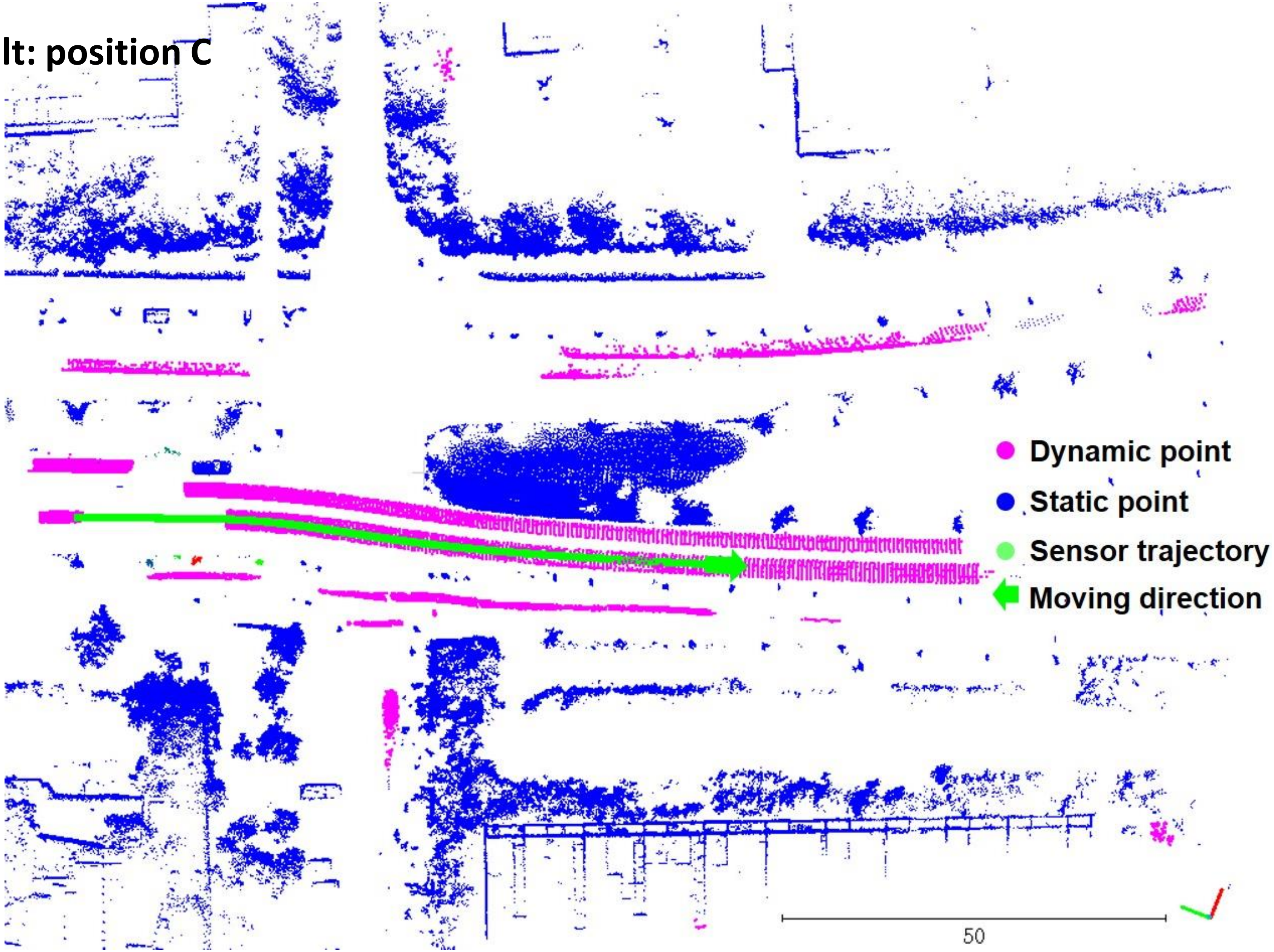
Implementation

Result: position B



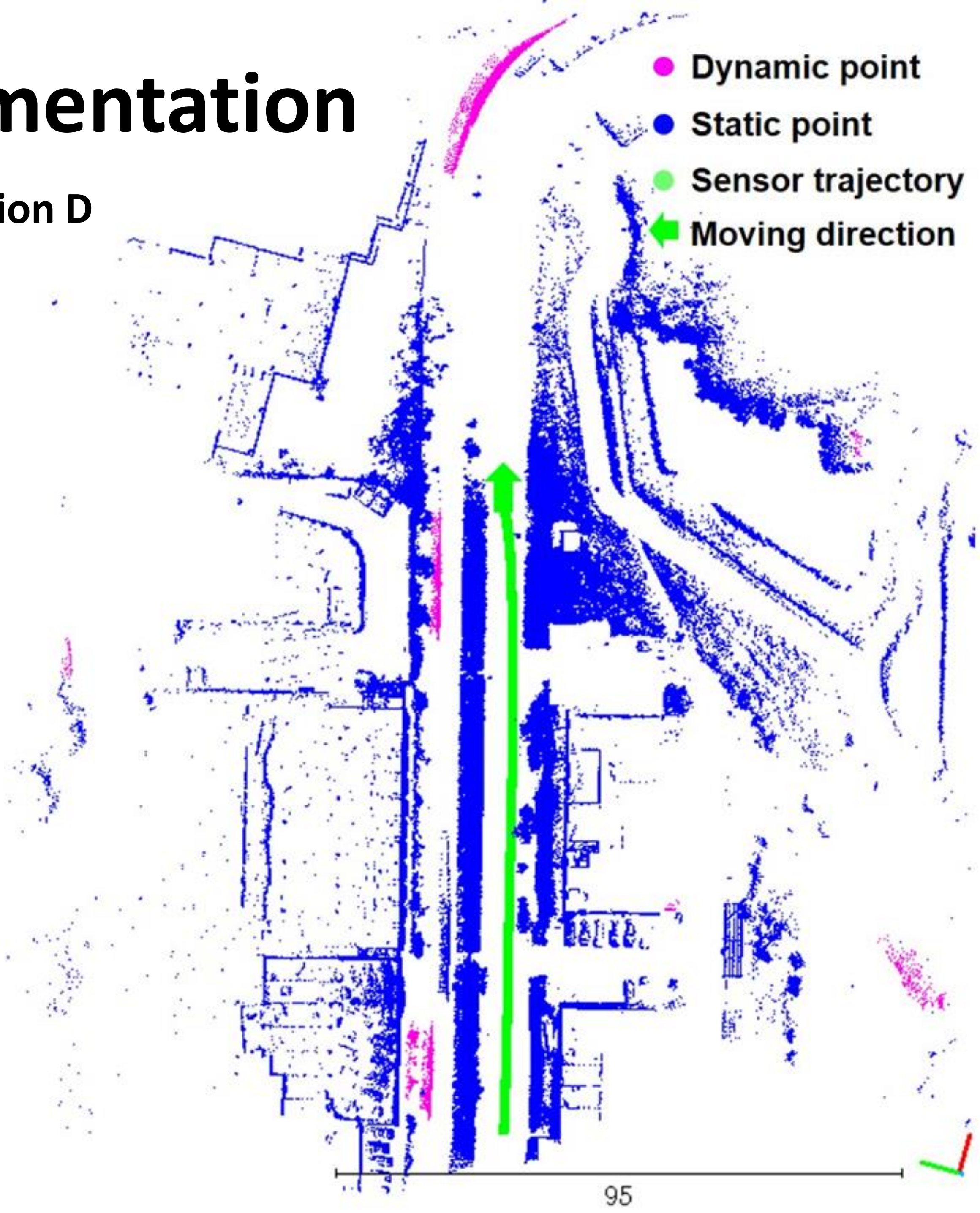
Implementation

Result: position C



Implementation

Result: position D



Implementation

Result: Accuracy Assessment

The weighted average producer's and user's accuracies for dynamic objects are **88.004%** and **82.624%,** respectively.

		Dynamic Points in Ground Truth	Static Points in Ground Truth	User's Accuracy (UA)
Position A	Dynamic Points in Implementation Result	57705	15408	78.926%
	Static Points in Implementation Result	188	3564668	99.995%
	Producer's Accuracy (PA)	99.675%	99.570%	
Position B	Dynamic Points in Implementation Result	11002	3597	75.361%
	Static Points in Implementation Result	1665	4600092	99.964%
	Producer's Accuracy (PA)	86.856%	99.922%	
Position C	Dynamic Points in Implementation Result	33529	1978	94.429%
	Static Points in Implementation Result	2234	4628689	99.952%
	Producer's Accuracy (PA)	93.753%	99.957%	
Position D	Dynamic Points in Implementation Result	6813	1951	77.738%
	Static Points in Implementation Result	2351	4673725	99.950%
	Producer's Accuracy (PA)	74.345%	99.958%	

The confusion matrix with the corresponding user's accuracies and producer's accuracies of the four case sites.

Implementation

Result: Accuracy Assessment

The weighted average overall accuracy is **99.833%**.

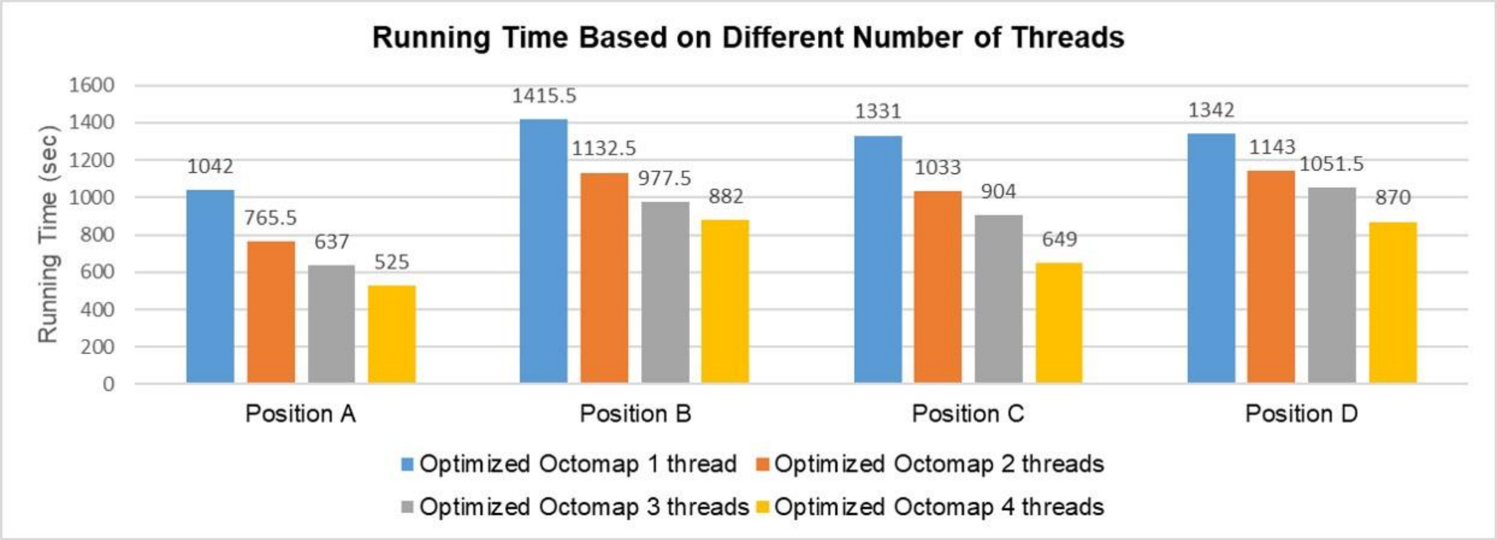
Position	Overall Accuracy (OA)	Kappa Coefficient (KC)
A	99.571%	0.878
B	99.886%	0.780
C	99.910%	0.936
D	99.908%	0.674
Weighted Average Value	99.833%	0.814

The overall accuracies and kappa coefficients of the four case sites

Implementation

Result: Running Time

The proposed method can be applied to parallel computing. Four threads are on average **79.545%** more efficient than a single thread.



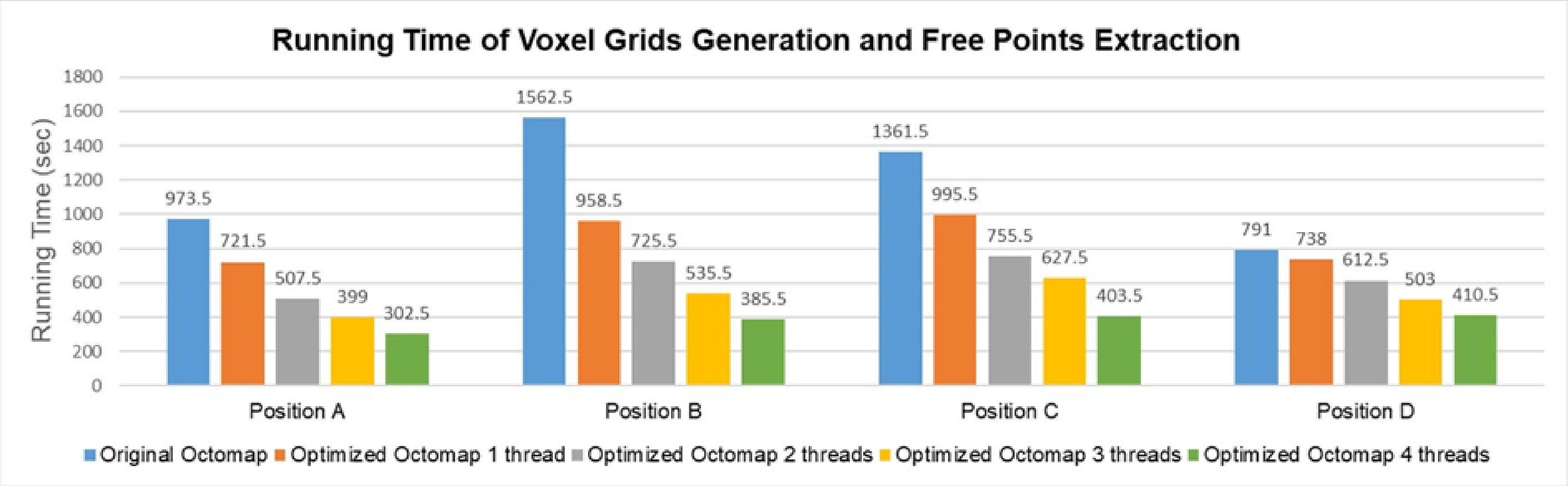
Case Site	Point Number	Optimized Octomap 1 Thread (sec)	Optimized Octomap 2 Threads (sec)	Optimized Octomap 3 Threads (sec)	Optimized Octomap 4 Threads (sec)	Total Speed-up
Position A	3637969	1042	765.5	637	525	98.476%
Position B	4616356	1415	1132	977.5	882	60.488%
Position C	4666430	1331	1033	904	649	105.085%
Position D	4684840	1342	1143	1051	870	54.253%

Running time based on different number of threads in four case sites

Implementation

Result: Comparison with the Original Octomap (Running Time)

The proposed method accelerates on average **35.472%** over the original Octomap with single thread enabled. This advantage is more obvious with more threads enabled.



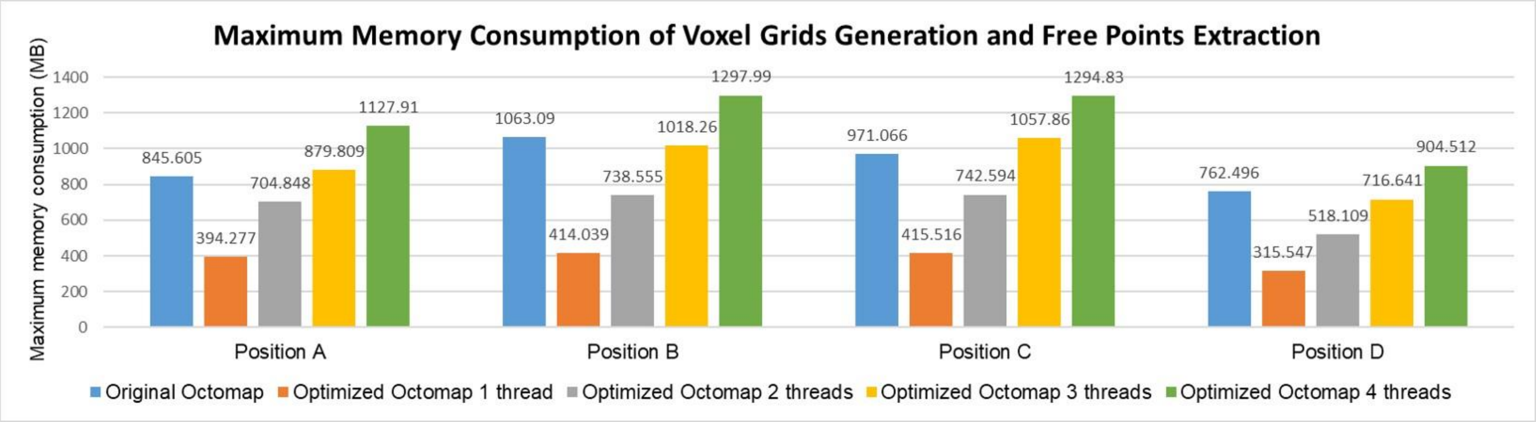
Case Site	1 thread	2 threads	3 threads	4 threads
Position A	34.927%	91.823%	143.985%	221.818%
Position B	63.015%	115.369%	191.783%	305.318%
Position C	36.765%	80.212%	116.972%	237.423%
Position D	7.182%	29.143%	57.256%	92.692%
Average Speed-up	35.472%	79.137%	127.499%	214.313%

The computational speed-up of the proposed optimized Octomap method compared to the original Octomap method.

Implementation

Result: Comparison with the Original Octomap (Maximum Memory Consumption)

The maximum memory consumption of the proposed optimized Octomap method is only on average **42.437%** of that of the original Octomap with a single thread. The memory consumption of the proposed method with three threads enabled is close to that of the original Octomap method.



Case Site	Optimized Octomap 1 thread	Optimized Octomap 2 threads	Optimized Octomap 3 threads	Optimized Octomap 4 threads
Position A	46.627%	83.354%	104.045%	133.385%
Position B	38.947%	69.472%	95.783%	122.096%
Position C	42.790%	76.472%	108.938%	133.341%
Position D	41.383%	67.949%	93.986%	118.625%
Average Values	42.437%	74.312%	100.688%	126.862%

The maximum memory consumption of the proposed optimized Octomap method as a percentage of the original Octomap.

Conclusion

Main Contribution

This thesis designs an Octomap-based dynamic object detection method for MLS data that is **compute-friendly**, **memory-friendly** and suitable for **parallel computing**.

This thesis defines the **local vehicle height restriction** as the upper boundary of the ROI, further narrowing the ROI to **reduce the computational cost** in following steps.

The method is tested with four case sites and its producer's and user's weighted average dynamic object detection and extraction accuracies of this method are **88.004%** and **82.624%**, respectively.

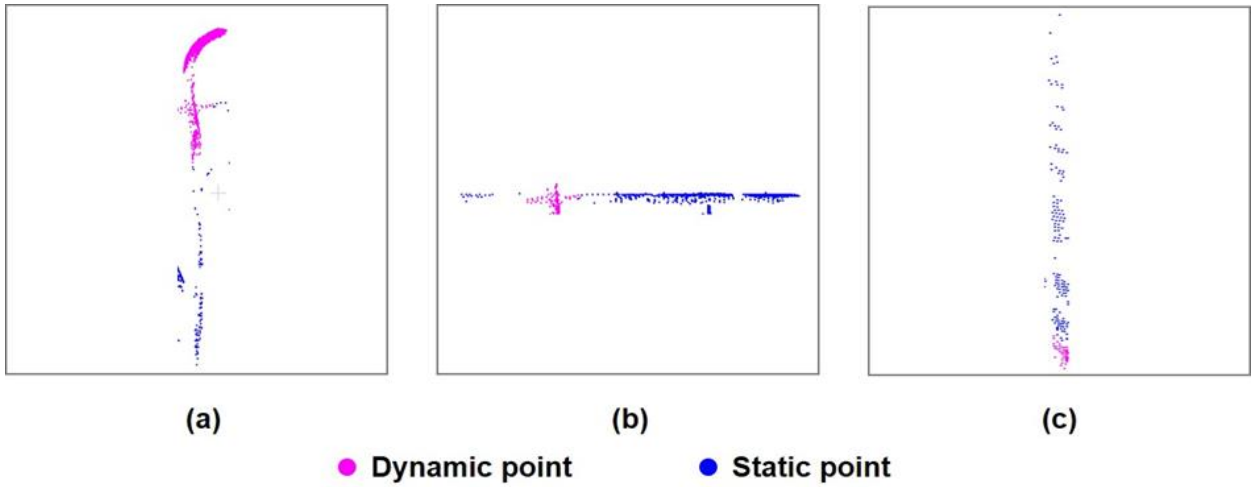
Conclusion

1. How to detect and remove dynamic objects and avoid residue?

First using spatial conflicts to extract free points from Octomaps. Then delimiting the ROI and removing noise and vegetation. Finally using KNN spatial search with filtered free points as seed points to extract complete dynamic objects and avoid residue.

The weighted average producer's accuracy for dynamic object detection is **88.004%**.

The very few incomplete results occur mainly on sparse objects far from the MLS sensor.



Sparse objects

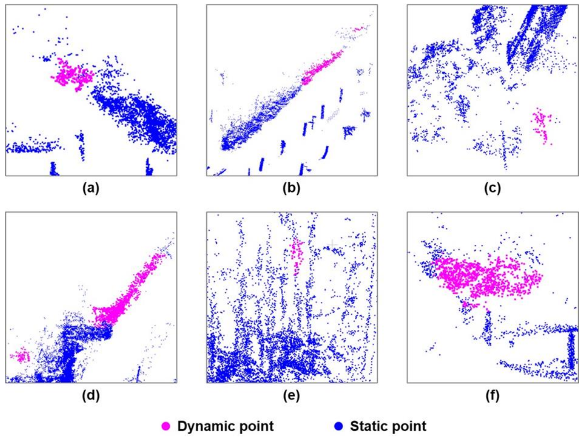
Conclusion

2. How to avoid detecting and removing static environment objects?

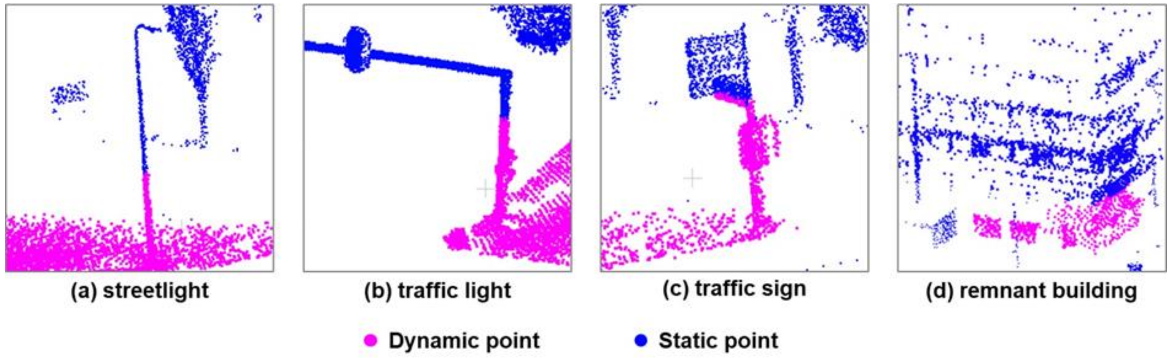
First delimits the ROI and then remove vegetation and noise in free points. So the final free points include only dynamic points as much as possible.

The weighted average user's accuracy for dynamic object detection is **82.624%**.

In the results, only a small number of static points, such as vegetation, remaining buildings, and pole-like objects are mis-detected.



Vegetation

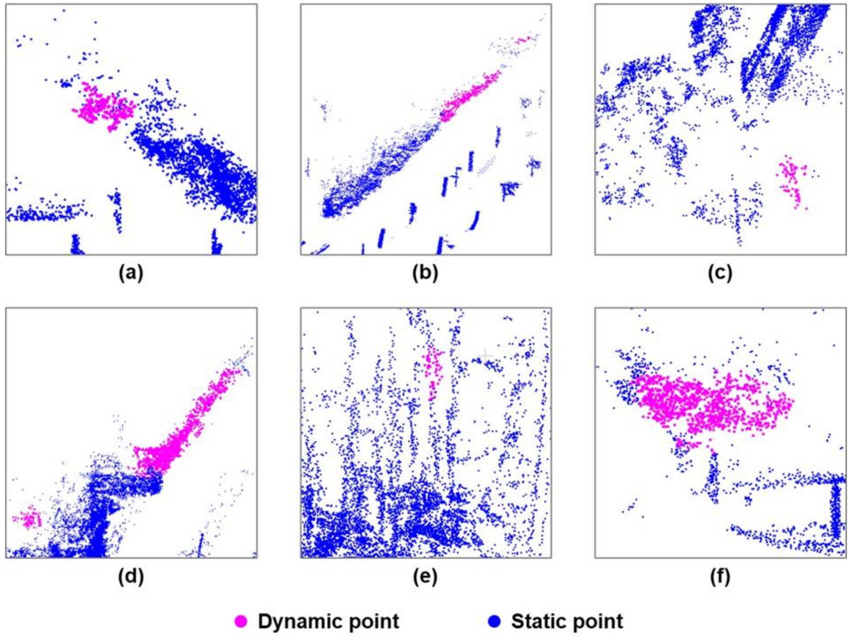


Remnant buildings and pole-like objects

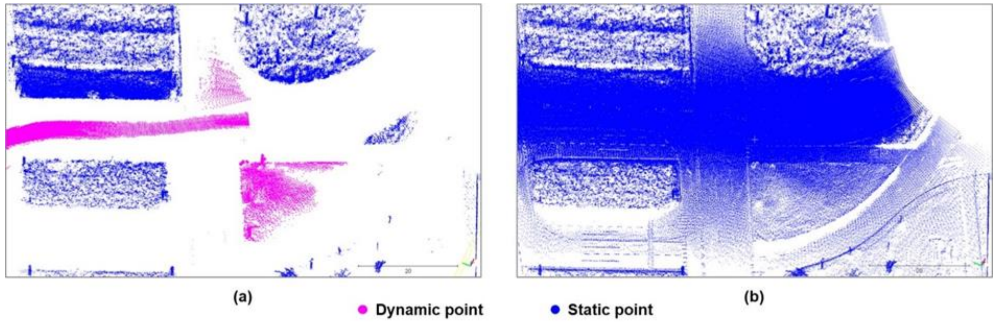
Conclusion

3. What factors affect the detection results?

The main cause of misdetection is the performance of the **vegetation and noise removal methods**. The minor cause is the performance of the **ground filtering method**.



Vegetation

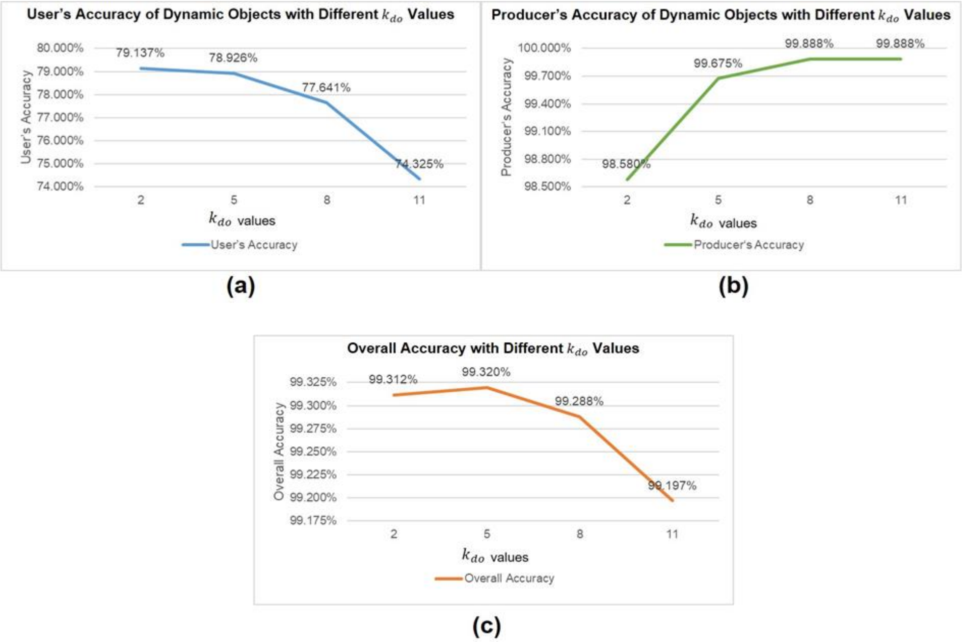


Ground

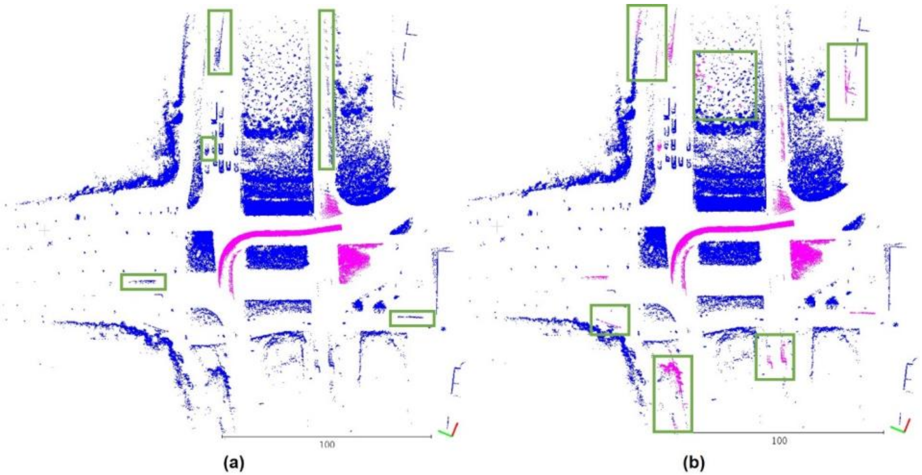
Conclusion

3. What factors affect the detection results?

The **choice of parameter values** also influences the final result. The number of nearest neighbors used to extract dynamic objects (k_{do}) is chosen as a test case. Changing k_{do} value directly affects the final accuracies.



The user's accuracy (a) and the producer's accuracy (b) of dynamic objects, and the overall accuracy (c) of the whole results using different k_do values.

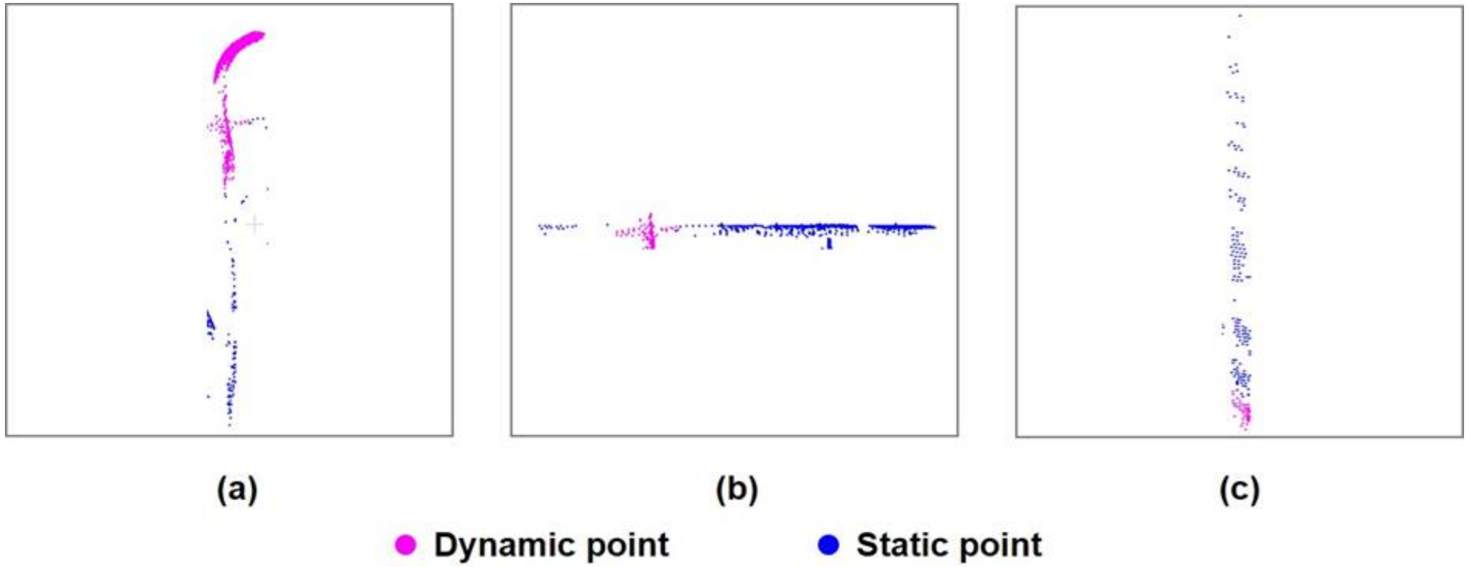


The detection results of position A when k_{do} is set to 2 (a) and 11 (b).

Conclusion

3. What factors affect the detection results?

The main reason why dynamic objects are not successfully detected is the **distance of the object from the MLS sensor**. A dynamic object that is too far from the MLS sensor causes its captured points to be too sparse and thus difficult to be detected completely. But this problem can be solved by adding neighbor data frames.

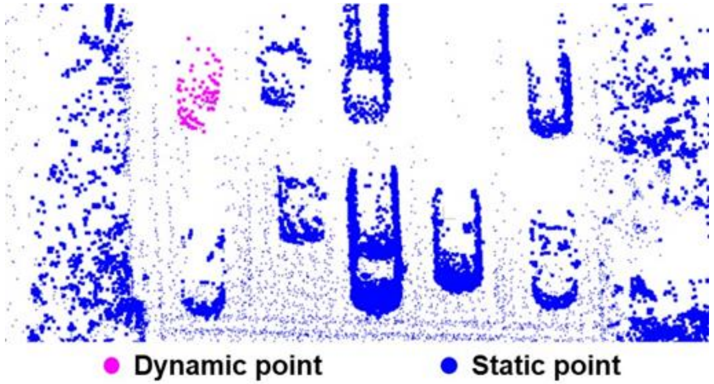


Sparse objects

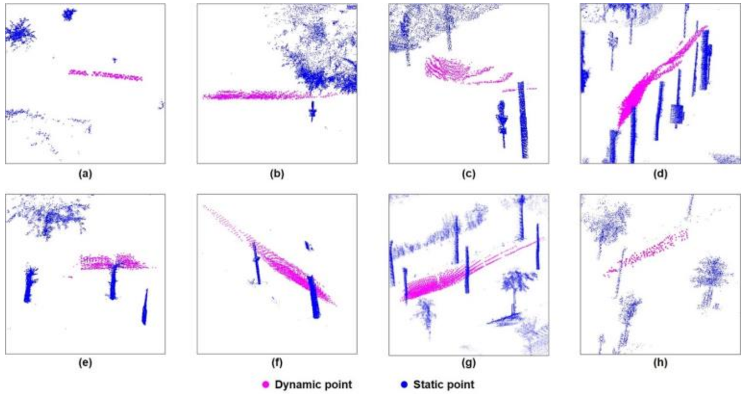
Conclusion

3. What factors affect the detection results?

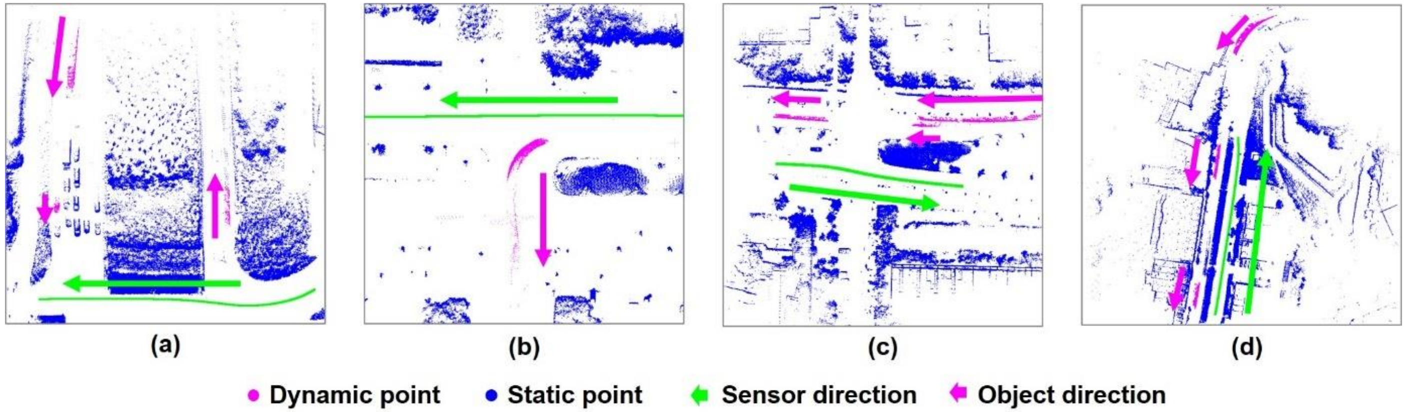
The **size**, **speed**, and **moving direction** of dynamic objects are not observed to have significant effects on the detection results in the implementation.



Braking vehicle



Bicycles



Objects moving in directions different from the MLS sensor

Conclusion

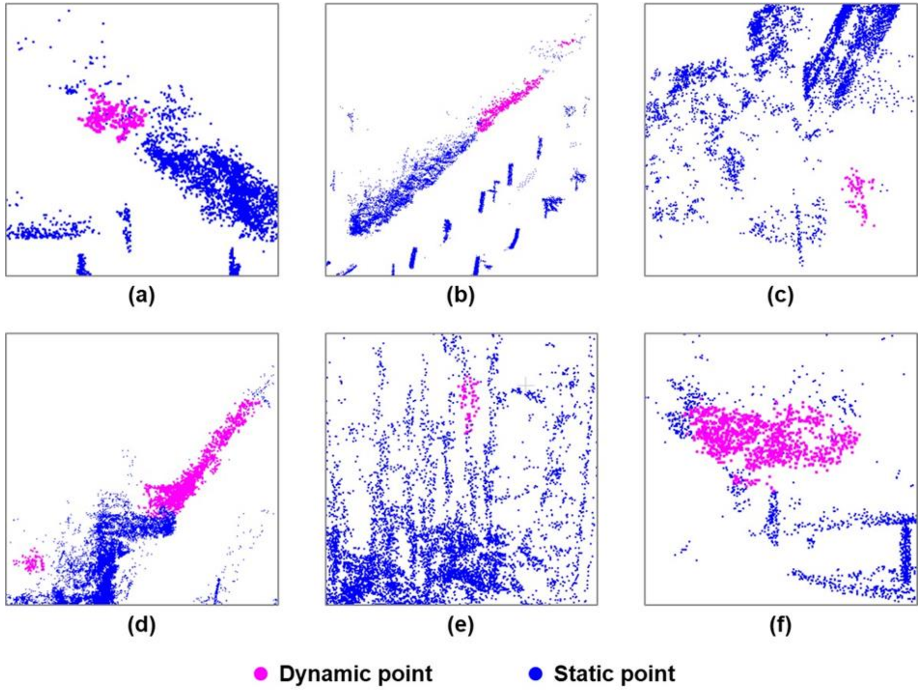
4. How to use MLS sensor trajectory to assist detection and removal operations?

- **Reconstruct the LiDAR rays** in Octomap with corresponding MLS capture points.
- **Delimit the upper and lower boundaries of the ROI** by obtaining the MLS sensor height from the trajectory.

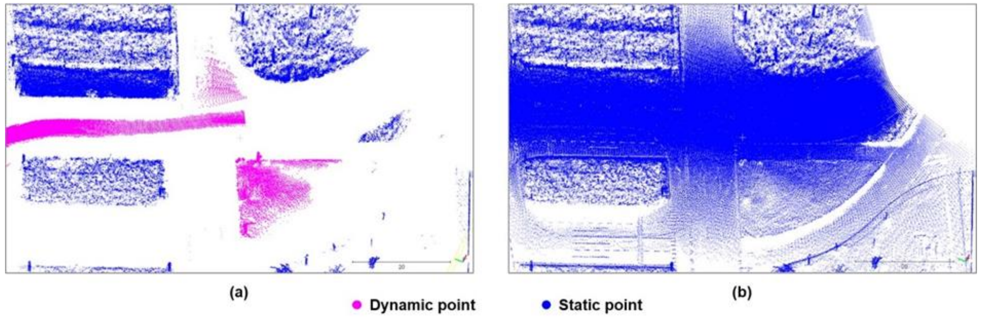
Conclusion

5. What types of objects often lead to misdetection?

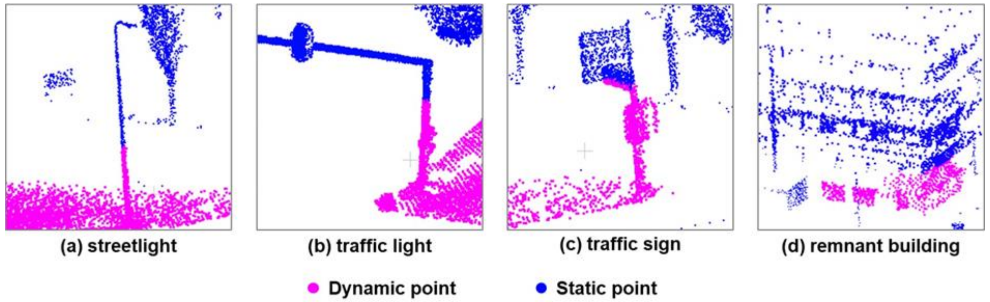
The most common mis-detected objects in the implementation results are **vegetation**, and other less common mis-detected objects include some small **ground areas**, **remnant buildings**, **pole-like objects** such as streetlights, traffic lights, and traffic signs.



Vegetation



Ground



Remnant buildings and pole-like objects

Conclusion

6. How to improve the computational efficiency for large-scale data?

- Reduce the computation and memory requirements by **generating multiple smaller Octomaps** to avoid generating a very huge Octomap.

- Obtain a smaller ROI by **removing the ground area and the high-altitude space** to reduce the computation cost in subsequent steps.

- Most steps of the whole processing workflow can be **accelerated with parallel computing**.

Future Work

- Self-adaptive threshold values
- Detection methods optimized for sparse dynamic objects
- Extension to more application scenarios
- Integration of better static object detection methods
- Speed detection and direction tracking
- More realistic LiDAR ray simulation
- Replacing the voxel grid with point-based structures
- GPU-accelerated computation
- Integration with DBMS

References

1. Efficient LiDAR-based Semantic SLAM. 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 4530–4537. https://doi.org/10.1109/IROS40897.2019.8967704
2. Dewan, A., Caselitz, T., Tipaldi, G. D., & Burgard, W. (2016). Motion-based detection and tracking in 3D LiDAR scans. 2016 IEEE International Conference on Robotics and Automation (ICRA), 4508–4513. https://doi.org/10.1109/ICRA.2016.7487649
3. Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., & Burgard, W. (2013). OctoMap: An efficient probabilistic 3D mapping framework based on octrees. Autonomous Robots, 34(3), 189–206. https://doi.org/10.1007/s10514-012-9321-0