sensors

Special Issue Reprint

Intelligent Point Cloud Processing, Sensing and Understanding (Volume II)

Edited by
Miaohui Wang and Sukun Tian

mdpi.com/journal/sensors

MDPI

Intelligent Point Cloud Processing, Sensing and Understanding (Volume II)

Intelligent Point Cloud Processing, Sensing and Understanding (Volume II)

Guest Editors

Miaohui Wang
Sukun Tian

MDPI

Basel • Beijing • Wuhan • Barcelona • Belgrade • Novi Sad • Cluj • Manchester

Guest Editors

Miaohui Wang
School of Information
Engineering
Shenzhen University
Shenzhen
China

Sukun Tian
School of Stomatology
Peking University
Beijing
China

Editorial Office
MDPI AG
Grosspeteranlage 5
4052 Basel, Switzerland

This is a reprint of the Special Issue, published open access by the journal *Sensors* (ISSN 1424-8220), freely accessible at: https://www.mdpi.com/journal/sensors/special_issues/ZW695MGH36.

For citation purposes, cite each article independently as indicated on the article page online and as indicated below:

Lastname, A.A.; Lastname, B.B. Article Title. *Journal Name* **Year**, *Volume Number*, Page Range.

ISBN 978-3-7258-4687-0 (Hbk)
ISBN 978-3-7258-4688-7 (PDF)
https://doi.org/10.3390/books978-3-7258-4688-7

Contents

About the Editors

Miaohui Wang

Miaohui Wang (Senior Member, IEEE) received a Ph.D. degree from the Department of Electronic Engineering, The Chinese University of Hong Kong (CUHK), Hong Kong, China. From 2014 to 2015, he was a researcher working on standardizing video coding at the Innovation Laboratory, InterDigital Inc., San Diego, CA, USA. From 2015 to 2017, he was a senior researcher studying computer vision and machine learning at The Creative Life (TCL) Research Institute of Hong Kong, Hong Kong, China. Currently, he is a tenured Associate Professor at the College of Electronics and Information Engineering, Shenzhen University (SZU), China. He has authored or coauthored 100+ peer-reviewed papers in top-tier international journals and conferences. Dr. Wang was the recipient of the Best Thesis Award from the Ministry of Education of Shanghai City and Fudan University (FDU), respectively. He received the First Prize of Sci-Tech Progress Award of Guangdong Province (2024) and the Outstanding Reviewer Award from IEEE International Conference on Multimedia & Expo (2021). He serves as an Associate Editor for *IEEE SIGNAL PROCESSING LETTERS*.

Sukun Tian

Sukun Tian received a Ph.D. degree in the manufacture engineering of aeronautics and astronautics from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2020. He became a Post-Doctoral Fellow with the School of Mechanical Engineering, Shandong University, Jinan, China, in 2023. He is currently an Assistant Professor/Associate Researcher and a Ph.D. Supervisor with the Center of Digital Dentistry, Peking University School and Hospital of Stomatology, Beijing, China. He has authored or coauthored over 40 peer-reviewed papers in journals and conferences. His current research interests cover a wide range of topics related to biomedical engineering, medical image analysis, intelligent manufacturing, and artificial intelligence (AI) techniques in healthcare and medicine applications.

sensors

Editorial

A Brief Introduction to Intelligent Point Cloud Processing, Sensing, and Understanding: Part II

Miaohui Wang [1,*] and Sukun Tian [2]

[1] Guangdong Key Laboratory of Intelligent Information Processing, College of Electronics and Information Engineering, Shenzhen University, Shenzhen 518052, China

[2] School of Stomatology, Peking University, Beijing 100081, China; sukhum169@bjmu.edu.cn

* Correspondence: wang.miaohui@gmail.com

1. Introduction

Point cloud, which represents the three-dimensional (3D) digital world, is one of the fundamental data carriers in many emerging applications [1], including the autonomous driving, robotics, and geospatial fields. Advancements in sensor technologies have facilitated the acquisition of point clouds from diverse platforms [2], perspectives, and spectra. This progress has underscored the necessity to address the generation [3], processing [4], analysis [5], and quality evaluation [6] of point cloud data.

The demand for intelligent point cloud processing and analysis has surged across various industries [7]. For examples, accurate 3D models enhance design, construction, and maintenance in construction and architecture [8], improving project outcomes and reducing costs. Integrating point cloud data into physical object systems ensures more reliable digital representations of structures [9]. In autonomous driving, real-time point cloud processing is critical for vehicle perception [10], where LiDAR sensors help vehicles to detect and navigate their environment, ensuring safety and reliability. Additionally, industries such as manufacturing, cultural heritage preservation, and urban planning rely on point cloud data for quality control, digital restoration, and city modeling. The incorporation of artificial intelligence (AI) has revolutionized point cloud applications [11], where AI-driven technology helps to automate object recognition, segmentation, and classification, significantly improving efficiency and accuracy [12].

This Special Issue serves as a comprehensive collection of recent advances in point cloud processing across various sensors. A total of ten contributions from the Republic of Korea, rhe UK, the PR China, Belgium, and the USA have been ultimately accepted for publication. These contributions delve into diverse aspects of point clouds, including dataset establishment, registration, detection, performance evaluation, receptive field analysis, and plant feature extraction. We provide a brief introduction to each of the collected contributions in the following section.

2. Overview of Contributions

Contribution 1 introduces an innovative framework to enhance LiDAR-based datasets for advanced driver-assistance systems (ADASs), improving the representation of distant objects by generating synthetic object points from real point clouds. The proposed framework consists of three key modules: (1) position determination identifies optimal locations and orientations for synthetic objects, employing ground filtering to separate ground and non-ground points; selects candidate positions; resolves collisions; and determines object

https://doi.org/10.3390/s25051310

poses; (2) object generation converts LiDAR data from Cartesian to spherical coordinates to generate synthetic points, where a spherical point-tracking technique and a "point wall" method mitigate excessive data loss, preserving object shape fidelity; (3) synthetic annotation automatically labels object points with attributes such as position, size, pose, and occlusion, ensuring consistency with datasets such as KITTI-360. Experimental results show that integrating these synthetic objects into training datasets enhances the performance of 3D detection models.

Contribution 2 advances 3D data processing by introducing a point cloud-specific attention mechanism to enhance convolutional neural networks (CNNs). Traditional CNNs struggle with point clouds due to its unstructured and unordered nature. To address these issues, the authors propose a channel attention mechanism tailored for point clouds, integrating it into the ConvPoint benchmark. The resulted module enhances feature emphasis, improving the network's focus on critical information. Experimental results show that incorporating the attention mechanism increases the mean intersection over union (mIoU) score, outperforming the base ConvPoint framework. This enhancement enables more effective processing of complex point clouds, with applications in autonomous driving, robotics, and virtual reality.

Contribution 3 proposes a novel algorithm for efficiently extracting phenotypic traits of rice plants using terrestrial laser scanning (TLS) data. It is noteworthy that traditional manual measurements are labor-intensive and time-consuming, so this study proposes an automated approach leveraging TLS data to extract key phenotypic features, including crown diameter, stem perimeter, plant height, surface area, volume, and projected leaf area. The extraction process employs several point cloud processing methods: (1) a neighborhood search algorithm calculates crown diameter and stem perimeter by establishing geometric relationships between point clouds; (2) an alpha-shape algorithm reconstructs the plant's 3D surface to determine surface area and volume; (3) extended hierarchical density-based spatial clustering is used to group plant stem point clouds and obtain the tiller number. Based on these algorithms, the study enhances the efficiency and accuracy of phenotypic data collection, offering a valuable resource for rice breeding and growth monitoring.

Contribution 4 introduces a flexible and adaptive framework that enhances feature learning through the innovative use of receptive field space (RFS) and attention mechanisms. Since traditional methods rely on manually defined local neighborhoods, they may be inflexible and may fail to capture both local details and global dependencies. To address these problems, the authors propose constructing an RFS mechanism that extracts effective features across multiple receptive field ranges, allowing adaptive scale selection for each point. Moreover, they develop an RFS attention method, which dynamically adjusts the network's focus across receptive field ranges, enhancing feature representation capability. This mechanism is integrated into a network architecture for point cloud classification and segmentation. Experimental results show that the proposed RFS effectively captures both local and global features, leading to improved 3D point cloud analysis.

Contribution 5 presents a comprehensive evaluation of six point cloud registration methods for aligning computer-aided design (CAD) models with real-world 3D scans. Unlike prior studies relying on synthetic datasets, this study utilizes point clouds from the Cranfield benchmark, incorporating CAD-sampled models and 3D scans of physical objects. The authors introduce real-world complexities such as noise and outliers, providing a more rigorous assessment. They evaluate three classical registration methods (i.e., GO-ICP, RANSAC, FGR) and three learning-based approaches (i.e., PointNetLK, RPMNet, ROPNet) using metrics such as recall, accuracy, computation time, and robustness to noise and partial data. This study can provide valuable findings that highlight the strengths and limitations of classical and learning-based registration techniques for real-world data, offering practical guidance for domain researchers in selecting methods based on accuracy, efficiency, and noise resilience.

Contribution 6 introduces a refined point cloud registration method that leverages geometric constraints and a dual-criteria evaluation process. It is noted that traditional point-to-point methods often suffer from inaccuracies due to erroneous matches and noise. The proposed approach enhances reliability by requiring only two correspondences (i.e., instead of the conventional three) to generate a transformation matrix, reducing computational complexity. Specifically, keypoints are detected to establish initial correspondences, with high-quality matches selected for improved alignment. Rotation and translation matrices are then computed using centroids and local reference frames. The optimal transformation matrix is determined based on the overlap ratio and inlier count. Experimental results demonstrate that integrating geometric constraints with a comprehensive evaluation strategy significantly enhances both accuracy and efficiency.

Contribution 7 presents an innovative LiDAR-based method for dynamic target detection, where motion states are evaluated by analyzing positional and geometric differences in point cloud clusters across consecutive frames. To accurately pair clusters representing the same target, a double registration algorithm is introduced, where a coarse registration is performed via iterative closest point (ICP) for initial pose estimation, followed by fine registration using random sample consensus and a four-parameter transformation for precise inter-frame alignment. This dual-step process standardizes coordinate systems, facilitating cluster association. Based on these paired clusters, the study constructs a classification feature system and employs the XGBoost decision tree for motion state evaluation. To improve training efficiency, a Spearman rank correlation-based bidirectional search reduces feature dimensionality, optimizing the classification subset. Meanwhile, a double Boyer–Moore voting–sliding window algorithm refines detection accuracy.

Contribution 8 introduces a robust solution for 3D object detection by effectively leveraging distance information and preserving critical point features, thereby enhancing the accuracy and reliability of scene understanding in complex environments. Specifically, the authors propose a set abstraction enhancement (SAE) network to address challenging issues raised by sparse and irregular point cloud data, which incorporates three key modules: an initial feature fusion module, a keypoint feature enhancement module, and a revised group aggregation module. By emphasizing distance information, the proposed network enhances the representation of distant objects, mitigating the decline in reflectivity that occurs with increased range. Moreover, it reinforces the intrinsic features of keypoints before they are combined with aggregated features, ensuring that essential semantic information is retained. Finally, the semantic coherence of the aggregated features improves the network's ability to differentiate between objects. Experimental results demonstrate that the integration of distance features and the enhancement of keypoint characteristics contribute to the more accurate detection of objects at varying distances, addressing the challenges posed by point cloud sparsity and reflectivity attenuation.

Contribution 9 advances point cloud registration by addressing challenges in low-overlap environments, where traditional methods struggle due to their reliance on abundant, repeatable keypoints for accurate correspondence extraction. As a result, the authors propose a graph convolutional attention-based robust point cloud registration network (RRGA-Net) to optimize correspondences among sparse keypoints through a multi-layer channel sampling mechanism and a template matching module. By forming patches through feature weight filtering, the proposed network captures more comprehensive contextual features, which is crucial for effective registration in low-overlap scenarios.

Moreover, the integration of self-attention mechanisms allows the network model to dynamically adjust weights based on relationships between points, enhancing the capture of both local and global features. Experimental results demonstrate that RRGA-Net exhibits robust performance, particularly excelling in low-overlap scenarios.

Contribution 10 presents a novel framework for accurately measuring cuboid and cylindrical objects using point cloud data from time-of-flight (ToF) sensors. ToF sensors often produce low-resolution, noisy data with self-occlusions and multipath interference, distorting object shape and size. To address these issues, the authors propose an enhanced superquadric fitting technique designed for noisy and incomplete point clouds. The proposed framework first performs ground plane rectification using fiducial markers to align the ground horizontally, followed by segmentation to isolate the related objects. A superquadric shape is then fitted using non-linear least squares regression. The proposed method is tested on objects of known dimensions placed on various surfaces, including aluminum foil, black/white posterboard, and black felt. Experimental results demonstrate that the enhanced superquadric fitting, particularly the bounding method, significantly improves accuracy, making this approach valuable for precise object measurement applications.

3. Conclusions

The evolution of point cloud acquisition and analysis has been propelled by AI advancements. This Special Issue compiles a diverse portfolio of contributions that address critical challenges in point cloud processing, sensing, and understanding. The selected studies push the boundaries of current knowledge by offering innovative solutions to existing challenges and unlocking new 3D applications. We anticipate that these developments can further expand the applications of point clouds across various industries, offering new opportunities and valuable insights for both researchers and practitioners to drive research and innovation in this field.

Author Contributions: Original draft preparation, M.W.; review and editing, M.W. and S.T. All authors have read and agreed to the published version of this manuscript.

Funding: This research received no external funding.

Acknowledgments: The authors express their sincere gratitude to the Journal Office of the MDPI journal *Sensors* for the kindness and assistance shown in the preparation of this Editorial.

Conflicts of Interest: The authors declare no conflicts of interest.

List of Contributions

1. Kim, K.; Lee, S.; Kakani, V.; Li, X.; Kim, H. Point Cloud Wall Projection for Realistic Road Data Augmentation. *Sensors* **2024**, *24*, 8144.
2. Umar, S.; Taherkhani, A. PointCloud-At: Point Cloud Convolutional Neural Networks with Attention for 3D Data Processing. *Sensors* **2024**, *24*, 6446.
3. Wang, K.; Pu, X.; Li, B. Automated Phenotypic Trait Extraction for Rice Plant Using Terrestrial Laser Scanning Data. *Sensors* **2024**, *24*, 4322.
4. Jiang, Z.; Tao, H.; Liu, Y. Receptive Field Space for Point Cloud Analysis. *Sensors* **2024**, *24*, 4274.
5. Denayer, M.; De Winter, J.; Bernardes, E.; Vanderborght, B.; Verstraten, T. Comparison of Point Cloud Registration Techniques on Scanned Physical Objects. *Sensors* **2024**, *24*, 2142.

6. Kang, C.; Geng, C.; Lin, Z.; Zhang, S.; Zhang, S.; Wang, S. Point Cloud Registration Method Based on Geometric Constraint and Transformation Evaluation. *Sensors* **2024**, *24*, 1853.
7. Xu, A.; Gao, J.; Sui, X.; Wang, C.; Shi, Z. LiDAR Dynamic Target Detection Based on Multidimensional Features. *Sensors* **2024**, *24*, 1369.
8. Zhang, Z.; Bao, Z.; Tian, Q.; Lyu, Z. SAE3D: Set Abstraction Enhancement Network for 3D Object Detection Based Distance Features. *Sensors* **2023**, *24*, 26.
9. Qian, J.; Tang, D. RRGA-Net: Robust Point Cloud Registration Based on Graph Convolutional Attention. *Sensors* **2023**, *23*, 9651.
10. Rodriguez, B.; Rangarajan, P.; Zhang, X.; Rajan, D. Dimensioning Cuboid and Cylindrical Objects Using Only Noisy and Partially Observed Time-of-Flight Data. *Sensors* **2023**, *23*, 8673.

References

1. Wang, M.; Yue, G.; Xiong, J.; Tian, S. Intelligent Point Cloud Processing, Sensing, and Understanding. *Sensors* **2024**, *24*, 283. [CrossRef] [PubMed]
2. Fang, J.; Zhou, D.; Zhao, J.; Wu, C.; Tang, C.; Xu, C.Z.; Zhang, L. LiDAR-CS dataset: LiDAR point cloud dataset with cross-sensors for 3D object detection. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Yokohama, Japan, 13–17 May 2024; pp. 14822–14829.
3. Zhou, C.; Zhong, F.; Hanji, P.; Guo, Z.; Fogarty, K.; Sztrajman, A.; Gao, H.; Oztireli, C. Frepolad: Frequency-rectified point latent diffusion for point cloud generation. In Proceedings of the Springer European Conference on Computer Vision (ECCV), Milan, Italy, 29 September–4 October 2024; pp. 434–453.
4. Fugacci, U.; Romanengo, C.; Falcidieno, B.; Biasotti, S. Reconstruction and preservation of feature curves in 3D point cloud processing. *Comput.-Aided Des.* **2024**, *167*, 103649. [CrossRef]
5. Zhou, X.; Liang, D.; Xu, W.; Zhu, X.; Xu, Y.; Zou, Z.; Bai, X. Dynamic Adapter Meets Prompt Tuning: Parameter-Efficient Transfer Learning for Point Cloud Analysis. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 17–21 June 2024; pp. 14707–14717.
6. Xie, W.; Liu, Y.; Wang, K.; Wang, M. LLM-guided Cross-Modal Point Cloud Quality Assessment: A Graph Learning Approach. *IEEE Signal Process. Lett.* **2024**, *31*, 2250–2254. [CrossRef]
7. Wang, M.; Huang, R.; Xie, W.; Ma, Z.; Ma, S. Compression Approaches for LiDAR Point Clouds and Beyond: A Survey. *ACM Trans. Multimed. Comput. Commun. Appl.* **2025**, 1–30. [CrossRef]
8. Choi, M.; Kim, S.; Kim, S. Semi-automated visualization method for visual inspection of buildings on BIM using 3D point cloud. *J. Build. Eng.* **2024**, *81*, 108017. [CrossRef]
9. Li, Y.; Xiao, Z.; Li, J.; Shen, T. Integrating vision and laser point cloud data for shield tunnel digital twin modeling. *Autom. Constr.* **2024**, *157*, 105180. [CrossRef]
10. Wang, M.; Huang, R.; Liu, Y.; Li, Y.; Xie, W. suLPCC: A novel LiDAR point cloud compression framework for scene understanding tasks. *IEEE Trans. Ind. Inform.* **2025**, 1–12. [CrossRef]
11. Zheng, Y.; Li, Y.; Yang, S.; Lu, H. Global-PBNet: A novel point cloud registration for autonomous driving. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 22312–22319. [CrossRef]
12. Zhu, Q.; Fan, L.; Weng, N. Advancements in point cloud data augmentation for deep learning: A survey. *Pattern Recognit.* **2024**, *153*, 110532. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

sensors

MDPI

Article

Point Cloud Wall Projection for Realistic Road Data Augmentation

Kana Kim [1], Sangjun Lee [2], Vijay Kakani [3], Xingyou Li [1] and Hakil Kim [1,*]

[1] Department of Electrical and Computer Engineering, Inha University, Incheon 22212, Republic of Korea
[2] EV Charger Development Team, Hyundai KEFICO Corp., Gunpo 15849, Republic of Korea
[3] Department of Integrated System Engineering, Inha Universiy, Incheon 22212, Republic of Korea
* Correspondence: hikim@inha.ac.kr

Abstract: Several approaches have been developed to generate synthetic object points using real LiDAR point cloud data for advanced driver-assistance system (ADAS) applications. The synthetic object points generated from a scene (both the near and distant objects) are essential for several ADAS tasks. However, generating points from distant objects using sparse LiDAR data with precision is still a challenging task. Although there are a few state-of-the-art techniques to generate points from synthetic objects using LiDAR point clouds, limitations such as the need for intense compute power still persist in most cases. This paper suggests a new framework to address these limitations in the existing literature. The proposed framework contains three major modules, namely position determination, object generation, and synthetic annotation. The proposed framework uses a spherical point-tracing method that augments 3D LiDAR distant objects using point cloud object projection with point-wall generation. Also, the pose determination module facilitates scenarios such as platooning carried out by the synthetic object points. Furthermore, the proposed framework improves the ability to describe distant points from synthetic object points using multiple LiDAR systems. The performance of the proposed framework is evaluated on various 3D detection models such as PointPillars, PV-RCNN, and Voxel R-CNN for the KITTI dataset. The results indicate an increase in mAP (mean average precision) by 1.97%, 1.3%, and 0.46% from the original dataset values of 82.23%, 86.72%, and 87.05%, respectively.

Keywords: LiDAR; point cloud; synthetic data; data augmentation; object detection

1. Introduction

The accurate perception of a vehicle's surrounding environment is essential for advancing autonomous driving technologies, as it enables the vehicle to safely navigate complex and dynamic road scenarios. Among various visual sensors, 3D LiDAR is particularly effective for capturing detailed spatial information, providing valuable insights into the position and distance of surrounding objects. However, acquiring large-scale, high-quality LiDAR data remains challenging due to the high cost of LiDAR sensors and the labor-intensive nature of 3D data labeling.

To address the demand for robust training data, recent synthetic data generation approaches have been explored, but they face key limitations: synthetic LiDAR point cloud generation, in most cases, demands intense computing power, and rendering-based methods struggle to represent sparse point clouds of distant objects accurately. Such limitations reduce the utility of synthetic data in training deep learning models for autonomous driving, particularly for scenarios involving complex interactions or specific orientations, such as vehicle platooning.

This study proposes a novel approach to generate synthetic object points from real point cloud data rather than synthetic data, improving the realism of distant object representations. Specifically, a "point-wall" technique is introduced to compensate for the excessive loss of details in distant objects, enhancing their shape fidelity. With the use of spherical point projection, the generated distant points are intended to resemble real

LiDAR point clouds since the synthetic points are modified from real LiDAR point clouds as opposed to other simulators. The notable difference between the real-world point clouds and the generated ones can be attributed to several aspects, including the return losses, which are not considered in the current study. Additionally, a pose determination module is integrated to capture realistic object orientations, enabling the representation of platooning vehicles on the road—an aspect previously unaddressed in virtual object generation.

The key contributions of this work are as follows:

- To enhance the representation of distant objects using the refinement of spherical point projection without the need for complex extrapolation techniques.
- To prevent excessive loss in virtual objects' details, ensuring the shapes of distant objects resemble real sensor data more closely via the point-wall method.
- To accurately depict the orientation of synthetic object points, supporting realistic platooning scenarios on roadways using the pose determination module.

The remainder of this paper is organized as follows. Section 2 outlines the research literature relevant to 3D object detection and data generation aspects. Section 3 describes the proposed methodology and its attributes, such as position estimation, object generation, and synthetic annotation. Section 4 provides information about the experimental environment, the dataset, synthetic LiDAR point cloud generation, deep learning model training, and relevant quantitative and qualitative results. Section 5 states the shortcomings of the proposed method and proposes potential future research based on the presented methodology, and Section 6 concludes the research study.

2. Related Works

Several methods are being developed in the context of LiDAR-based vehicle perception to improve detection performance, mainly by restoring lossy point cloud data to synthetic points. The main aim of these methods is to create synthetic points to enhance data, owning to a loss of 3D point cloud data and a lack of objects. However, because these methods require considerable computation, they are difficult to operate in an embedded environment. Data generation using well-designed simulators is an another way to overcome the shortage of point cloud data. A few methods, such as that proposed by Esmoris et al. [1], tackle this by exploring virtual laser scanning (VLS), demonstrating that simulated data can achieve comparable results to real-world data. VLS represents a scalable and cost-effective alternative, although further integration with dynamic scenarios is needed for its real-world application. VLS methods are often referred to as approaches to generating synthetic point clouds, including the method used in this study. Also, Beltran et al. [2] obtained dense point cloud data using LiDAR sensors and generated point cloud data for the desired rendering environments, such as simulators. There is a large focus on developing a method for collecting and using LiDAR data from 3D games [3] or a method for rendering and using data from the LiDAR sensor of an autonomous driving simulator platform [4,5]. The simulator's LiDAR sensor data are ideal as they lack real noise, so the rendered results exhibit a large difference compared with the real data. Also, synthetic LiDAR point cloud generation techniques using the GAN [6] model can be used as augmentation techniques for datasets. In addition, Yin et al. [7] enhanced the DART model with Monte Carlo-based methods, enabling precise satellite LiDAR simulations and facilitating data fusion with other remote sensing modalities. Furthermore, Yin at al. [8] extended LiDAR waveform simulation to multi-pulse systems and introduced new methods for simulating photon-counting data and solar noise in diverse configurations. Gastellu et al. [9] introduced a comprehensive 3D radiative transfer model that simulates the interaction between Earth, the atmosphere, and sensor specifications for remote sensing applications. Also, Gastellu et al. [10] expanded on DART by integrating LiDAR waveform simulation, demonstrating its versatility across various landscape and atmospheric configurations and its ability to model multi-scattering effects. Similarly, Yang et al. [11] presented DART-Lux, a novel LiDAR modeling approach that enhances simulation efficiency using a bidirectional path-tracing algorithm, while comparing different tracing methods for improved accuracy. Further Yang et al. [12] validated the DART-Lux model using real GEDI and ICESat2 data, quantifying inconsistencies in height measurements and incorporating atmospheric effects for more accurate LiDAR simulations in large-area landscapes.

LiDAR-Aug [13] was developed to generate synthetic object point clouds by generating point cloud data from synthetic objects. It expresses the point distribution of the real object's data but is vulnerable when generating distance points for the real object data in their version of synthetic point clouds. To address this, Xiao et al. [14] proposed SynLiDAR, a large-scale synthetic dataset with annotated point clouds, and the Point Cloud Translation (PCT) method to bridge the gap between synthetic and real point clouds. While effective in improving transfer learning strategies on 32 custom-built semantic classes, it still faces challenges in generating data for rare object categories because of data imbalances (little to no samples in the rare classes). Xiang et al. [15] built on this by introducing a data augmentation method using generative models like L-GAN to enhance rare classes in LiDAR point cloud datasets. This method effectively balances class distribution, improving recognition performance across both minority and majority classes. However, while generative models address data imbalance, they still rely on high-quality real data for model training.

In contrast, D-Aug [16] retrieved objects and integrated them into dynamic scenarios, taking into account the continuity of these objects across successive frames. However, D-Aug suffers from post-insertion occlusion due to complicated and cluttered situations arising after the object's integration into the LiDAR scenes. Zhang et al. [17] utilized a conditional generative model that employs segmentation maps as a guiding tool to ensure the accurate generation of adverse effects, significantly improving the robustness of perception and object detection systems in autonomous vehicles under diverse and challenging conditions. Although robust LiDAR segmentation [18] employs domain-specific augmentation methods such selective jittering to address complicated spatial interactions in varied weather situations, it faces issues in preserving dataset quality and computational needs. Text3DAug represents a scalable LiDAR data augmentation method [19]. The prompting system generates annotated 3D instances from written descriptions and automates augmentation without intense labeling.

The most recent studies on LiDAR simulation techniques highlight a convergence of methodologies aimed at improving efficiency, realism, and applicability across various domains. Lopez et al. [20] focused on GPU-based LiDAR simulation to generate dense semantic point clouds for deep learning (DL), offering remarkable speed improvements and scalability in procedural environments. Building on such foundational simulations, Winiwarter et al. [21] introduced HELIOS++, a modular framework capable of simulating diverse LiDAR scenarios, such as terrestrial and airborne scanning, emphasizing the balance between computational efficiency and physical realism. While these studies emphasize physical modeling, Anand et al. [22] explored physics-informed deep learning by incorporating incidence angles to improve LiDAR intensity predictions using the U-NET and Pix2Pix architectures. Extending these advancements, Zyrianov et al. [23] presented LidarDM, a novel latent diffusion model that generates 4D layout-aware LiDAR sequences, revolutionizing virtual scene generation for autonomous driving. Complementing these innovations, Eggert et al. [24] leveraged game engines to create synthetic point clouds for industrial object detection, bridging the gap between real and simulated data. Together, these studies contribute towards versatile, scalable, and high-fidelity LiDAR simulation frameworks tailored to emerging applications in robotics, remote sensing, and AI.

Table 1 summarizes the current state-of-the-art LiDAR synthetic point cloud-generating methodology and its essential characteristics.

Table 1. Insights into object generation and data augmentation for 3D object detection networks for LiDAR data.

Research Study	Methodology	Key Aspects	Limitations
Esmoris et al. [1]	Trains models with virtual laser scanning data	Automated scene and model training	Limited to specific real data applications
Yin et al. [7]	Extended DART model with Monte Carlo methods for satellite LiDAR simulation	Efficient scattering models; supports data fusion with other sensors	Focuses on vegetation and urban scenes
Zhao et al. [16]	LiDARsim uses real data, ray casting, and neural networks	Realistic LiDAR for autonomous testing	High fidelity needed, weather simulation issues
Zhang et al. [17]	Conditional generative model with segmentation maps	Large dataset, domain adaptation strategies.	Cannot generalize to more severe settings
Park et al. [18]	Generative models for long-tail object recognition	Generative augmentation for minority classes.	Focuses only on object recognition
Reichardt et al. [19]	Virtual laser scanning for semantic segmentation	Automated training, real-world data reliance.	Accuracy gaps, dynamic scene challenges
Lopez et al. [20]	GPU-based LiDAR simulator generates dense semantic point clouds for DL training	High speed (99% faster); large-scale labeling; procedural scene generation	Limited to procedural or static environments
Winiwarter et al. [21]	HELIOS++ simulates terrestrial, airborne, and mobile LiDAR with modular scene modeling	Handles vegetation, supports Python, has fast runtime, and creates training data	Slightly less accurate than ray-tracing models
Anand et al. [22]	Physics-informed DL for LiDAR intensity simulation using U-NET and Pix2Pix architectures	Adds incidence angle as input; improves intensity prediction accuracy	Lacks material property integration
Zyrianov et al. [23]	LidarDM generates realistic, layout-aware, temporally coherent 4D LiDAR sequences	4D generation; driving scenario guidance; high realism for simulations	Not real time; no intensity modeling yet
Eggert et al. [24]	Synthetic point cloud generation using Unreal Engine for object detection	High-quality clouds; suitable for industrial datasets	Sparse datasets; lacks specific real-world equivalency
Manivasagam et al. [25]	Simulates LiDAR with real data, simulations, and ML	Realistic LiDAR simulations	Requires large datasets, domain challenges
Fang et al. [13]	Rendering-based LiDAR augmentation	Point distribution representation	Low performance for long-distance objects
Xiao et al. [14]	SynLiDAR dataset creation, translation via PCT	Large synthetic dataset, transfer learning	Focused on segmentation, overfitting risk
Xiang et al. [15]	Generative models for LiDAR object recognition	Synthetic point clouds, minority class focus	Limited generalization, needs tailored models
Li et al. [26]	PointAugment auto-optimizes point cloud samples via adversarial learning	Sample-specific augmentation; improves shape classification	Limited to certain transformations

Figure 1 presents an overview of the proposed framework. The framework proposed in this study operates in three parts: position determination, spherical point projection, and synthetic annotation modules. In the position determination module, the position and pose of the synthetic object to be generated are determined. The spherical point projection module generates synthetic object points using the spherical point projection and point-wall methods. In the synthetic annotation module, labels are attached to the projected points. The input object contains the point model, which uses an open-source point cloud library to convert each triangular polygon in the 3D shape of the freely distributed .obj file into a surface composed of points.

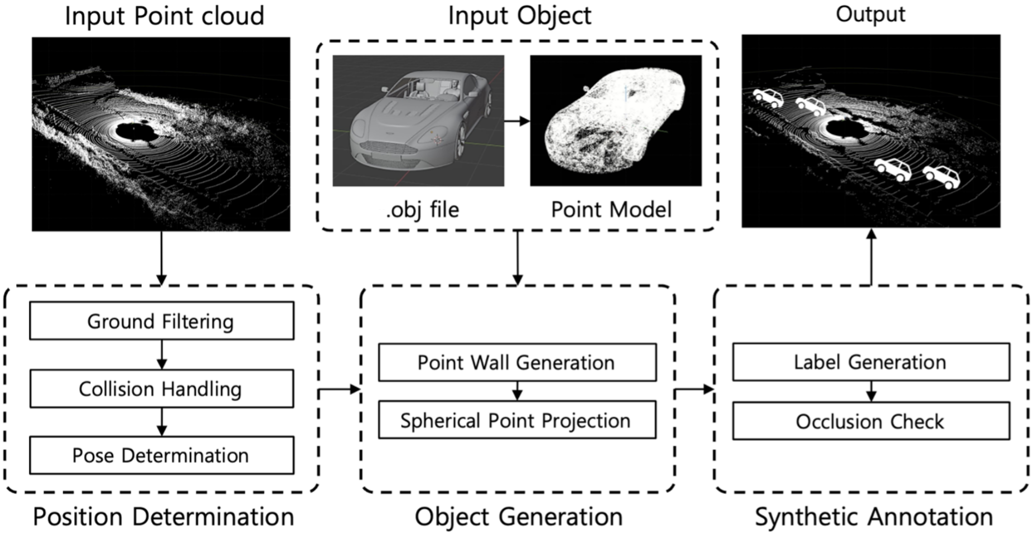


Figure 1. Overview of proposed framework.

3. Proposed Method

3.1. Position Determination

3.1.1. Ground Filtering

The position determination module determines the position and orientation of the synthetic object to be generated. Ground filtering is used to separate the input data into ground and non-ground data, and random coordinates within the ground data are set as candidate positions for generation. Subsequently, collision handling is performed between these position candidates, and the orientation of the object at the final chosen position is determined. Figure 2 shows the main algorithm of the position determination module.

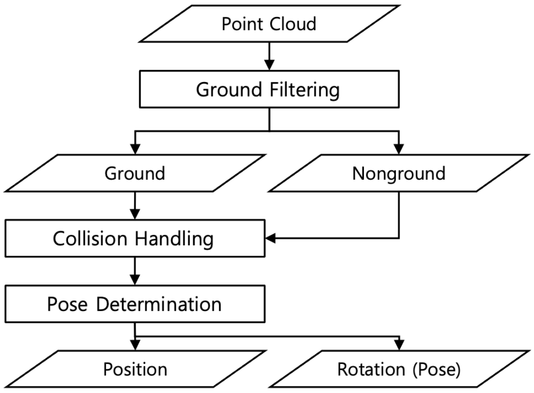


Figure 2. Main algorithm of position determination module.

This ground-filtering algorithm is used to determine an area in the input data where the synthetic object could be generated. This study uses the PatchWork++ [27] algorithm, which recognizes the ground by calculating the plane angle of a specific area. Subsequently, the ground data are randomly selected from the desired number of points and assigned as candidate positions for generation. Figure 3 shows the results of the ground-filtering algorithm.

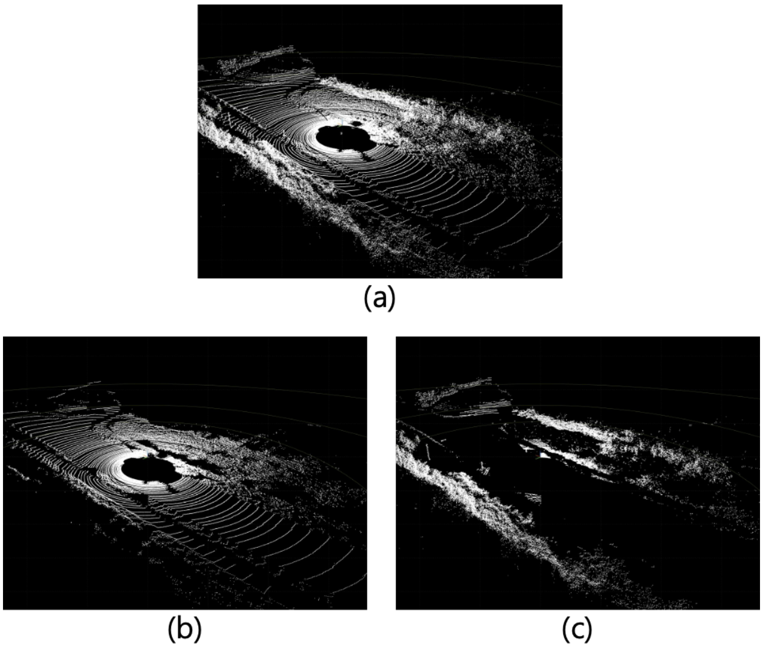


Figure 3. Ground filtering: (**a**) input LiDAR data; (**b**) filtered ground data of input point cloud; (**c**) filtered non-ground data.

3.1.2. Collision Handling

Collisions between synthetic objects (virtual objects) and real points (such as ground or non-ground points) are detected through a two-step process:

- Collision between virtual objects: After generating candidate coordinates for the synthetic objects (from the region of interest, or RoI, which is the ground point cloud), the algorithm first ensures that virtual objects do not overlap. It does so by selecting one candidate coordinate at random and removing all other coordinates within a certain distance, known as the "collision threshold". This ensures that virtual objects are spaced out properly to prevent overlaps as shown in Figure 4b.
- Collision with non-ground points: Once the first collision detection (between virtual objects) is completed, the remaining candidate coordinates are checked for collisions with non-ground points (such as vegetation, sidewalks, etc.). This is carried out by comparing the coordinates with the non-ground point cloud, which was previously segmented using a ground segmentation algorithm. If any candidate coordinates are too close to non-ground points (within a specified distance), they are discarded. For the second step, the collision threshold used for virtual object-to-non-ground collision detection is half the value used for virtual object-to-virtual object collisions. This ensures a finer level of collision avoidance when checking proximity to non-ground features like vegetation or sidewalks as shown in Figure 4c. This approach of using distance-based thresholds for collision handling can be less precise when compared to synthetic mesh-based pruning. Although a mesh-based approach would improve placement precision, the computational trade-offs may not justify its use in large-scale dataset generation, particularly when speed and scalability are prioritized. Therefore, this limited approach comprising distance-based thresholds for collision handling is utilized, which will be replaced by mesh-based approaches in the future.

3.1.3. Pose Determination

The pose determination algorithm determines the orientation of the synthetic object to be generated. This study uses the yaw value to determine the orientation; this parameter represents the rotation angle around the Z-axis in the 3D coordinate system and indicates the direction that the object is facing and the direction in which the vehicle is driving. The pose of the object is determined based on its position in the input data, considering the Korean road traffic infrastructure environment, and the following algorithm (Algorithm 1), which describes vehicles going straight, reversing, turning left, and turning right, as shown in Figure 4d. The pose determination module modifies the pose of the synthetic object by dividing the input point cloud space area by O_y, which is the lateral distance of the synthetic object O from the sensor (or the ego vehicle).

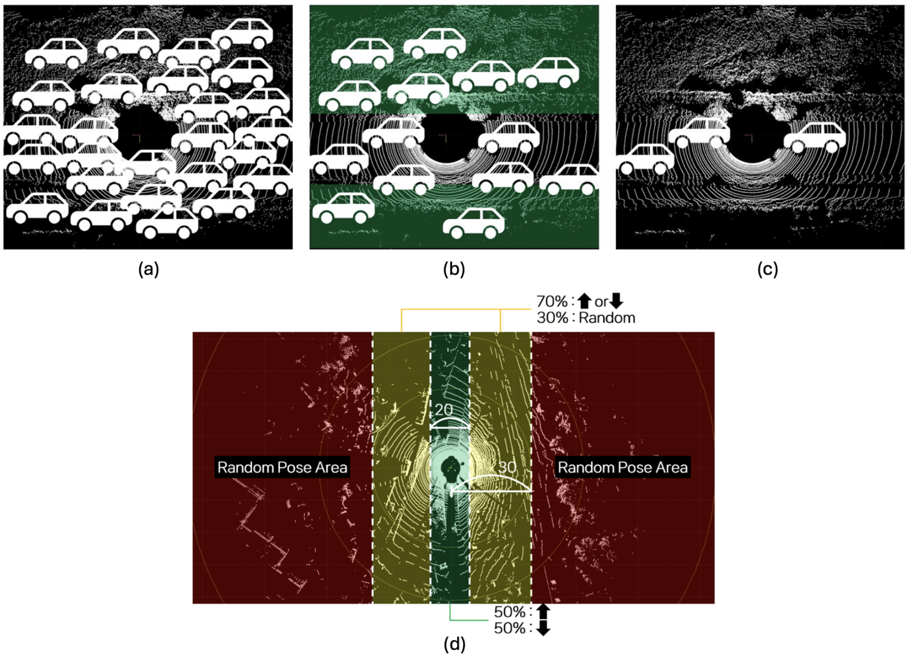


Figure 4. Collision handling and pose determination: (**a**) before collision handling; (**b**) after collision handling between virtual objects; (**c**) after collision handling between virtual objects and a non-ground point cloud; (**d**) vehicle pose distribution with respect to pose decision areas.

The yaw angle of synthetic objects (virtual objects) is determined through a multi-step process, taking into account the object's location, its class (vehicle or non-vehicle), and the surrounding real-world vehicles. This function can be expressed by Equation (1):

$$O_{Yaw} = \begin{cases} 0 \text{ or } \pi, & \text{if } |O_y| < 10 \\ 0 \text{ or } \pi \text{ or } Yaw_{rand}, & \text{else if } 10 < |O_y| < 30, \\ Yaw_{rand} & \text{else} \end{cases} \tag{1}$$

The yaw angle calculation is a two step process:

- Pose decision area: The initial yaw value is set based on the object's Y-coordinate (O_y) in the LiDAR sensor's coordinate system, divided into three areas:
  - Straight pose area ($O_y < 10$): the yaw angle is set to 0 (same direction) or π (opposite direction), chosen randomly.
  - Intersection pose area ($10 < O_y < 30$): the yaw angle is set to 0, π, or a random value (Yaw_{rand} between 0 and 2π).
  - Random pose area ($O_y \geq 30$): the yaw angle is set randomly between 0 and 2π.
- Update with nearby real vehicles: Once the yaw angle is assigned based on the pose decision area, it can be updated based on the orientation of nearby real vehicles. The input point clouds in this context are typically labeled with object categories such as "car", "truck", "pedestrian", etc. If a real vehicle is within a certain proximity to the synthetic object, the yaw value of the virtual object is updated to match the yaw angle of the nearest real vehicle. Thereby, the yaw value is determined with reference to the position and orientation derived from the bounding box. This step reflects the

real-world phenomenon where vehicles on the road often drive in the same direction (or opposite directions) in clustered groups, such as on highways or in dense traffic.

3.2. Object Generation

3.2.1. Spherical Point Cloud Projection

The SPCP module is responsible for generating synthetic object points by projecting real-world data (LiDAR point clouds) into synthetic models. This process involves several key steps:

- Coordinate transformation: The module first converts the input LiDAR point cloud data from Cartesian (orthogonal) coordinates to spherical coordinates. This step is necessary for applying the spherical point-tracking technique.
- Spherical point tracking: Once the data are in spherical coordinates, the module uses spherical point tracking and point cloud wall creation techniques to generate a synthetic model of the virtual object. This process defines the structure of the synthetic object based on the real-world point cloud data.
- Projection and final transformation: After applying the spherical point-tracking method, the resulting synthetic model is converted back into Cartesian coordinates. This step finalizes the projection of the virtual object into the synthetic point cloud data, effectively augmenting the original data with the new object.
- Integration into synthetic annotation: The augmented point cloud data are then passed into the synthetic annotation module, which processes the data further to fit the required data format, reflecting object occlusion and other relevant information like object type, position, and orientation.

Table 2 presents the data structures involved in generating synthetic object points by projecting real-world data into synthetic models. The proposed method uses the real acquired points to form synthetic object points. As a result, this work can more fully reflect the noise and loss distribution of real LiDAR sensors, and this is prominent for distant objects. Figure 5 shows the data before and after applying the proposed method.

Table 2. Data structures involved in projecting real-world data into synthetic models.

Values	Category	Data	Type
1	Class	Describes the type of object	string
3	Position	3D object location in LiDAR coordinates (in meters) Ex. [Pos_x, Pos_y, Pos_z]	float[]
3	Rotation	3D object rotation in LiDAR coordinates Ex. [0, 0, Yaw]	float[]
3	Dimension	3D object dimensions in LiDAR coordinates (in meters) Ex. [Dim_x, Dim_y, Dim_z]	float[]
1	Occluded	Integer (0,1,2,3) indicating occlusion state: 0 = fully visible 1 = partly occluded 2 = largely occluded 3 = unknown	int

There are several types of losses in LiDAR point cloud data; however, the excessive loss mentioned here refers to the loss caused by rays radiating from the sensor that do not reflect off any object. In general, LiDAR sensors such as Velodyne HDL-32E manufactured by Velodyne Lidar, Inc., San Jose, CA, USA , with a horizontal resolution of 0.08°∼1.33° and a vertical resolution of 1.33°, have a detection range of approximately 100 m, a horizontal field of view of 360°, and a vertical field of view of 41.33°. If there are no objects within 100 m, the radiated rays do not return. This type of loss needs to be compensated for, as it would not have occurred if the synthetic object had originally been in that position. Figure 6 shows a case in which an excessive loss occurs in the shape of a synthetic object.

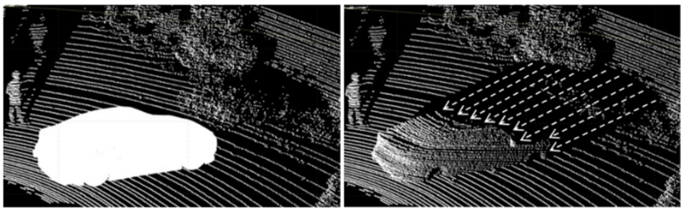


Figure 5. Spherical point projection method: white silhouette represents synthetic object onto which the real LiDAR points are projected (depicted as arrows).

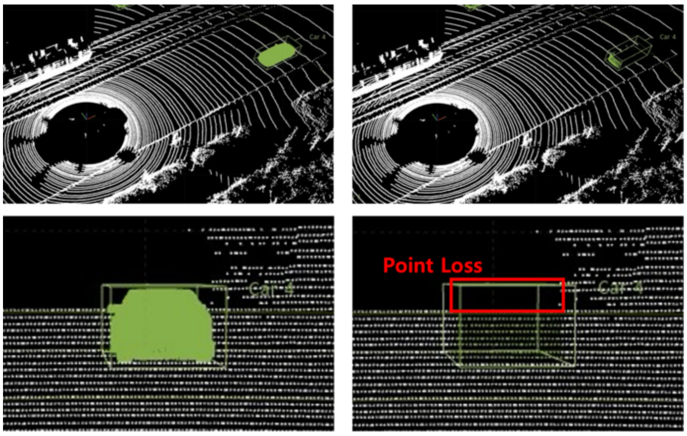


Figure 6. Point loss due to detection range of LiDAR sensor: green silhouette represents detected vehicle (car) when it is in the LiDAR's detection range and point loss (red box) when car is out of the LiDAR's detection range.

Algorithm 1 Spherical Point Projection Algorithm

```
Input: Input Point Cloud $P$, Inserted Synthetic Object $O$, Generated Point Wall $W$
  and Point Cloud with Synthetic Object $P'$
Output: Point cloud with synthetic object $P'$
Convert $P$, $O$ and $W$ from orthogonal coordinate system $(x, y, z)$ to spherical
  coordinate system $(r, \theta, \varphi)$.
for each point in $w$ in $W$ do
    for each point in $o$ in $O$ do
        if there is a point $p$ which has same $\theta$ and $\varphi$ with $w$ then
            $p_r \leftarrow o_r$
        else
            $w_r \leftarrow o_r$
Delete the rest of $o$ and $w$ return
```

3.2.2. Point Wall

A point wall was considered to compensate for this type of loss by generating a point cloud wall with a resolution that matched the performance of the LiDAR sensor with the input data to fill in the lost parts. This allows for excessive losses that would not have occurred if the synthetic object had been in that position to be compensated for, and distant objects can be represented more realistically. The point walls reflect the horizontal, res_h, and vertical, res_v, resolution of the input point clouds. The indices, i and j, of the point wall are calculated within the O_θ and O_φ ranges of the synthetic object data, as shown in Equation (2)

$$\begin{aligned} 0 \le i \le \frac{max(O_\theta)-min(O_\theta)}{res_h} \\ 0 \le j \le \frac{max(O_\varphi)-min(O_\varphi)}{res_v}, \end{aligned} \tag{2}$$

and to consider the scanning pattern of a LiDAR sensor, the resolutions and ranges in both the horizontal and vertical directions are parameterized along with the distance between the sensor and the point wall. Relevant parameters, such as $W_r = 100$ (maximum distance range of a Velodyne HDL-32E LiDAR sensor) and horizontal and vertical spacings, are specified with respect to the spherical coordinate system values (W_θ and W_φ) of the point wall, as shown in Equation (3). The width of the curved point wall depends on the horizontal field of view (FoV) of the LiDAR sensor, as specified in Table 3 and shown in Figure 7b.

$$\begin{aligned} W_\theta &= min(O_\theta) + (res_h \times i) + \epsilon_h \\ W_\varphi &= min(O_\varphi) + (res_v \times j) + \epsilon_v \\ W_r &= 100 \end{aligned} \tag{3}$$

are calculated using this index. Next, the horizontal noise and vertical noise, ϵ_h and ϵ_v, are added, similarly to the noise of the real sensor. Figure 7 shows the recovery of the loss of the synthetic object using a point wall. Inaccurate placement of the point wall can be avoided by using the input parameters of the sensor. The noise level in the position of the point cloud data is within 30% of the horizontal and vertical resolutions, which mitigates the possibility of generating duplicate point clouds in the same location. However, this study does not consider return losses according to the reflected intensity, which demands further investigation regarding materials and reflectivity.

Table 3. Performance of Velodyne HDL-64E and HDL-32E LiDAR sensors manufactured by Velodyne Lidar, Inc., San Jose, CA, USA.

		Velodyne HDL-64E (KITTI 360 Dataset)	**Velodyne HDL-32E (nuScenes Dataset)**
Range		∼120 m	∼100 m
Resolution	Horizontal	0.08°	0.08∼1.33°
	Vertical	0.4°	1.33°
Field of View	Horizontal	360°	360°
	Vertical	26.8°(−24.8∼+2)	41.33°(−30.67∼+10.67)

3.3. Synthetic Annotation

The synthetic annotation module generates labeling information for the generated synthetic object points. Labeling data for 3D objects typically include information such as the object’s position, size, pose, and occlusion and are automatically generated using information about the generated synthetic object, as shown in Figure 8.

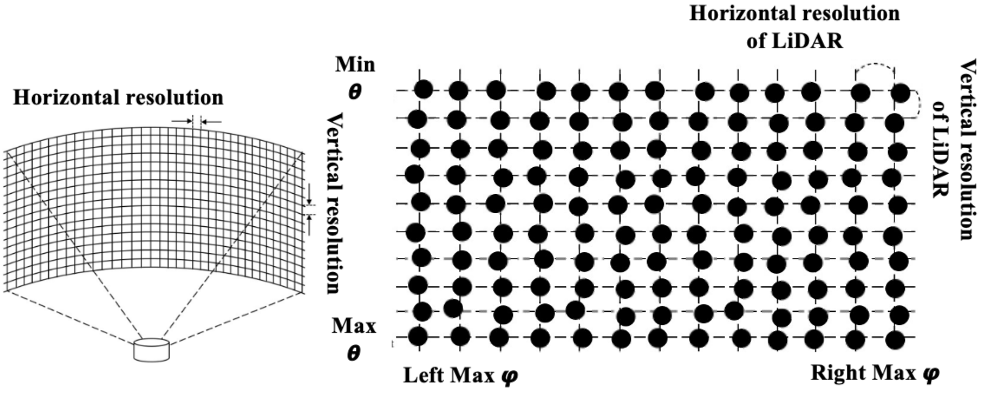


Figure 7. *Cont.*

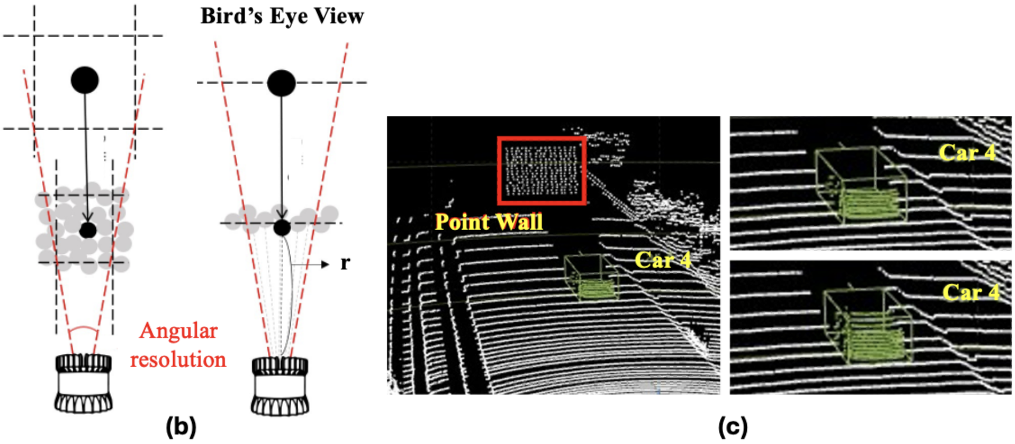


Figure 7. Point loss compensation by the point wall. (**a**) Appearance of the curved point wall generated by the proposed technique; (**b**) process of searching for point coordinates of a synthetic object point cloud model corresponding to a point in the input data and the arrows represent the perspective of normal view, bird's eye view; (**c**) point loss compensation for the synthetic object point generation.

Additional occlusion handling is performed because existing real objects can be affected by synthetic object points. For example, if a synthetic car is generated in front of a real car, it is occluded, which must be reflected in the labeling data. Through this process, the labeled data for synthetic object points were recorded and saved as a text file.

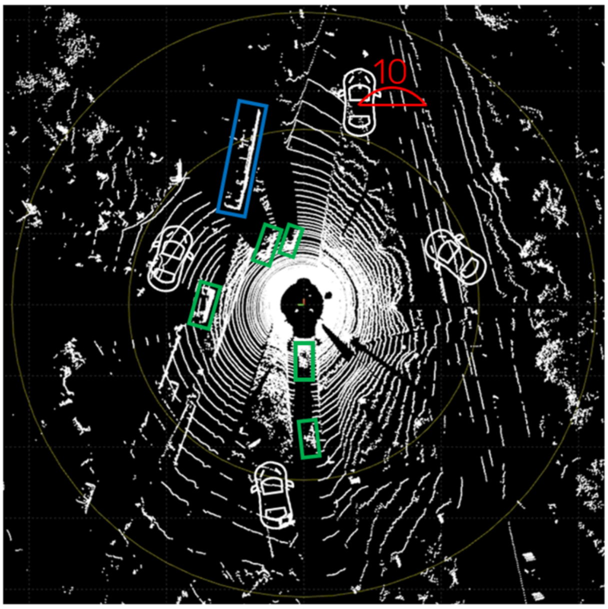


Figure 8. Adapting object rotation and position based on proximity and default parameters: green represents car objects, blue represents bus objects and red area indicates a horizontal range of ±10 m within which the synthetic cars (white cars) are generated.

The labeling data consist of the same format as the KITTI 360 dataset, which is the most commonly used 3D object detection dataset. Since the KITTI 360 dataset includes data in which 2D data and 3D data are fused, it is characterized by storing 3D location information data as 2D data. It basically includes the location, size, and direction data of a 3D cuboid and stores data on how much the object is hidden from other objects and how much it is cut off by the sensor's field of view. Table 4 shows the format of the KITTI 360 dataset's labeling data.

The 'truncated' and 'occluded' aspects of the synthetic object are calculated through occlusion handling, and information about the synthetic object generated in the data is obtained before the synthetic object is created, which is then modified when the real object is affected by the synthetic object to reflect this. The completed labeling data are written and stored as a .txt format file, just like the labeling file in the KITTI 360 dataset.

Table 4. Three-dimensional object detection labeling data format for Kitti 360 dataset.

Values	Name	Description
1	Type	Describes the type of object: 'Car', 'Van', 'Truck', 'Pedestrian', 'Person_sitting', 'Cyclist', 'Tram', 'Misc' or 'Dont Care'
1	Truncated	Float from 0 (non-truncated) to 1 (truncated), where truncated refers to the object leaving the image boundaries
1	Occluded	Integer (0,1,2,3) indicating occlusion state: 0 = fully visible 1 = partly occluded 2 = largely occluded 3 = unknown
1	Alpha	Observation angle of object, ranging [−pi, pi]
4	Bbox	2D bounding box of object in the image (0-based index) : contains left, top, right, and bottom pixels
3	Dimensions	3D object dimensions: height, width, and length (in meters)
3	Location	3D object location x, y, z in camera coordinates (in meters)
1	Rotation_y	Rotation, ry, around Y-axis in camera coordinates [−pi, pi]

4. Experimental Results

4.1. Experimental Environment

Dataset and Model

For the synthetic LiDAR point cloud generation experiment, the KITTI 360 [28] and nuScenes [29] datasets were used. The KITTI 360 dataset is one of the most commonly used autonomous driving datasets and includes 3D LiDAR data. The LiDAR sensor used for data acquisition was the HDL-64E model developed by Velodyne, which is a 64-channel model. The nuScenes dataset is an autonomous driving dataset that includes LiDAR data and uses the HDL-32E model from Velodyne, a 32-channel LiDAR. These two datasets, shown in Table 3, were chosen for the experiment because they were acquired using LiDAR sensors with channels different from those of Velodyne. The datasets include object bounding boxes and AP3D (%) evaluation metrics, assessing object detection performance across easy, moderate, and hard scenarios, considering occlusion and object size.

For the deep learning model training experiment, PointPillars [30], PV-RCNN [31], and Voxel R-CNN [32] models were used as 3D object detection models. The PointPillars model is a network that shows outstanding computational speed owing to its pillar-shaped feature extraction and is still widely used due to its real-time performance. Both the PV-RCNN and Voxel R-CNN models achieved state-of-the-art (SOTA) results, demonstrating superior performance compared to previous research.

4.2. Synthetic LiDAR Point Cloud Generation

The experiment was divided into a car class generation experiment, which accounted for the majority of the experiment, and pedestrian and cyclist class generation experiments. In addition, a generation experiment was conducted to verify the representation of platooning, as shown in Figure 9, using the synthetic object position determination module of the proposed framework and the data distribution with object distance, as shown in Figure 10.

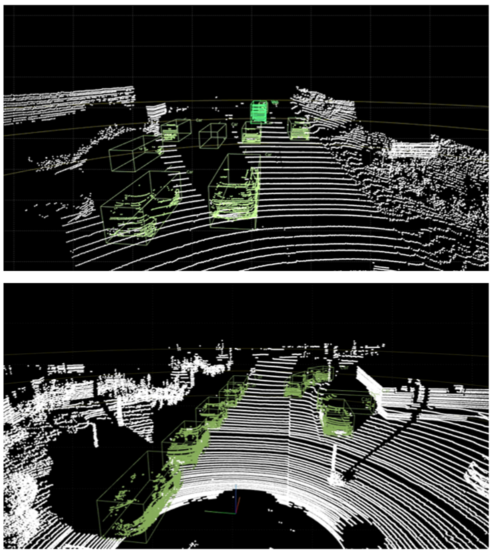
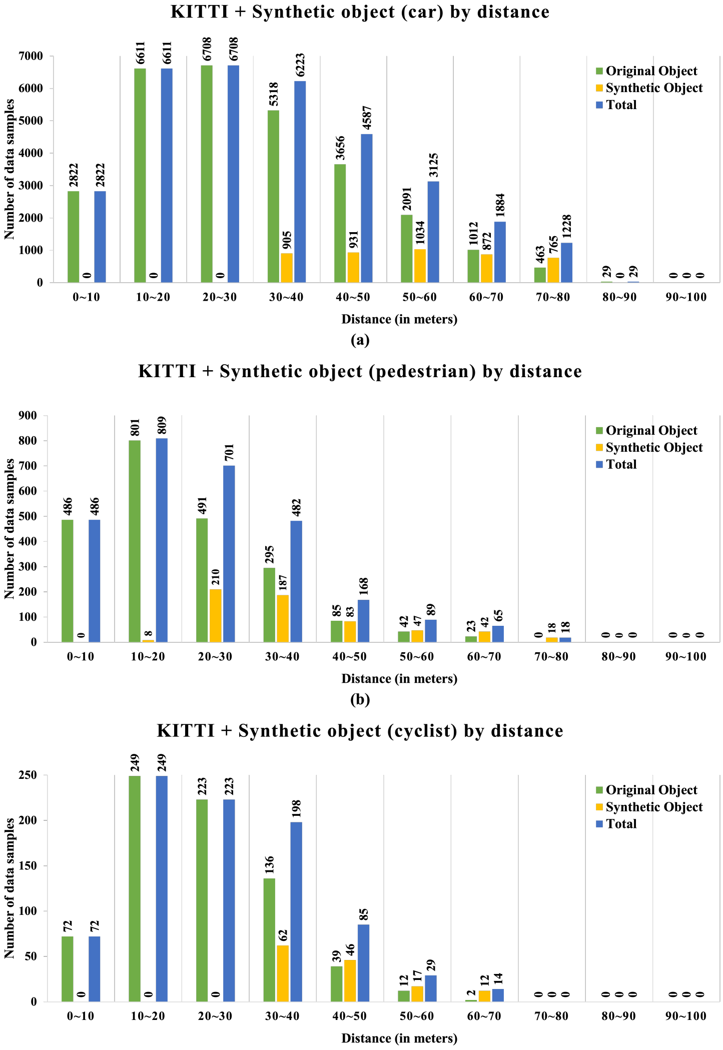


Figure 9. Platooning situation that appears in the KITTI 360 dataset: green represents car objects, and light green represents van objects.



(a)

(b)

(c)

Figure 10. The proportion of data generated by each category—(**a**) car, (**b**) pedestrian, and (**c**) cyclist—at different distances from the original data.

4.2.1. Car Class

The car class represents the most frequently observed objects in autonomous driving datasets. The experiment was divided into near distances of over 15 m and far distances of over 50 m, generating synthetic object points on both the 64-channel LiDAR data from the KITTI 360 dataset and the 32-channel LiDAR data from the nuScenes dataset. Figure 11 shows the resulting image of the synthetic LiDAR point cloud generation for the car class.

The results of creating synthetic car class objects using the proposed framework showed that it is capable of generating synthetic object points with shapes similar to those of real objects for both near and far distances. The shape of objects in LiDAR point cloud data varies depending on the channel and the performance of the sensor. This experiment proved that these variations could be significantly well represented by the shape of the objects. Figure 12 compares the shapes of the generated synthetic car objects with those of the real objects at the same distance. Synthetic LiDAR point cloud generation occurred at distances greater than 50 m and showed the appearance of real vehicles at similar distances as the synthetic vehicles generated by LiDAR-Aug and the proposed method. The LiDAR-Aug method was implemented based on the proposed method, and synthetic object points reflecting Gaussian noise were generated. Figure 13 compares the shapes of the synthetic object points generated by the existing LiDAR-Aug and proposed frameworks. The experimental results confirmed that our method represents long-distance objects that are more similar to real ones.

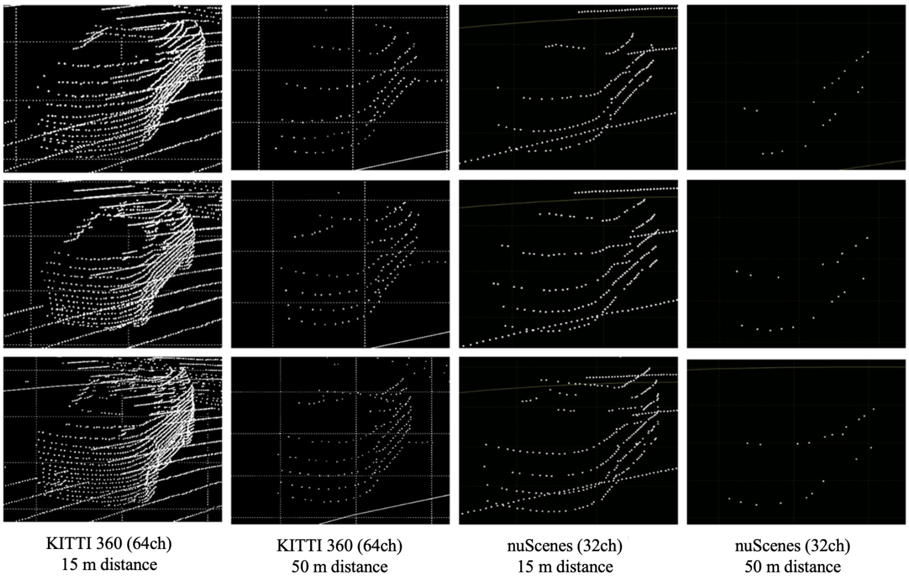


Figure 11. Results of synthetic LiDAR point cloud generation experiment for car class.

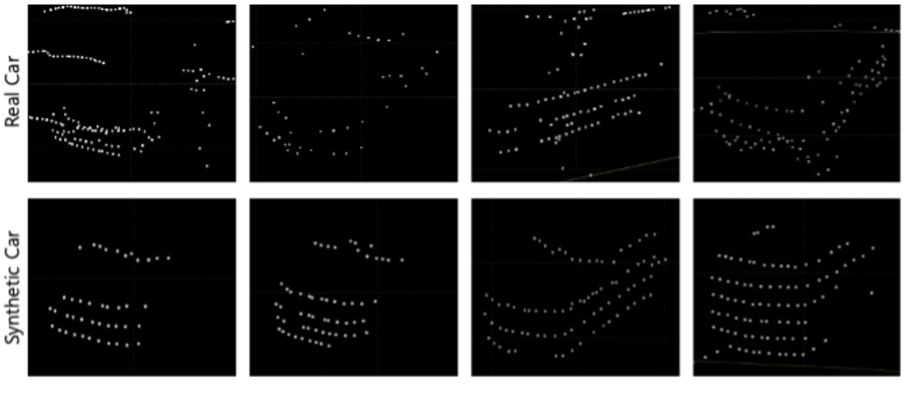


Figure 12. Comparison of distant synthetic car objects generated using the proposed method with real distant car objects.

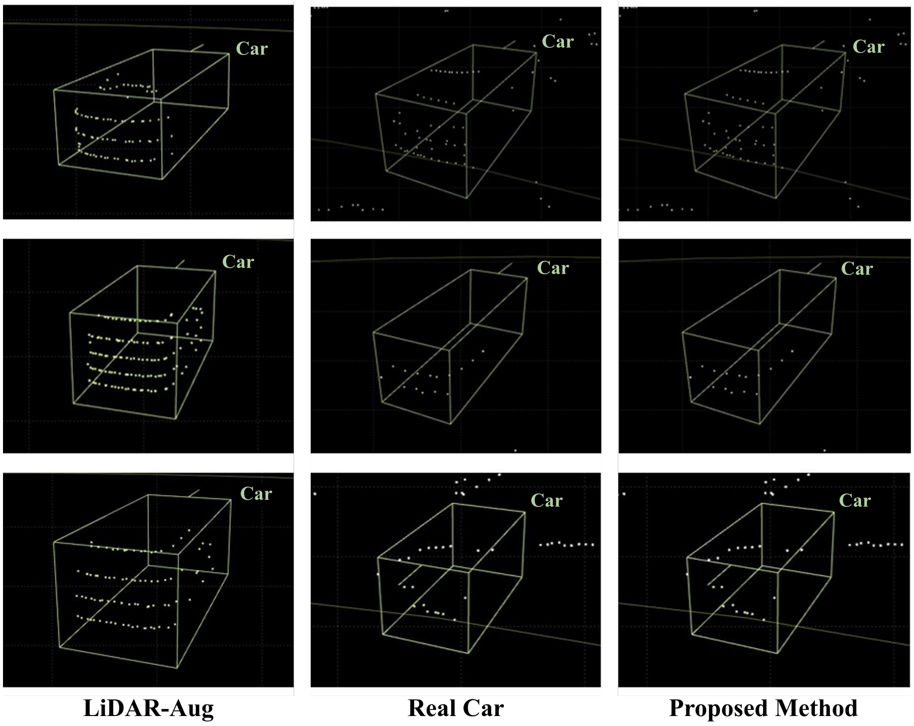


Figure 13. Comparison of realism factor between LiDAR-Aug and the proposed method.

4.2.2. Pedestrian and Cyclist Classes

This study also conducted generation experiments for the pedestrian and cyclist classes, which are the most used classes after the car class in autonomous driving datasets. Similarly to the car class, synthetic object points were generated at near distances of 15 m and far distances of 50 m, and it was confirmed that the synthetic object points generated at each distance showed shapes similar to those of real objects. Although the shapes of the objects were not as distinct as those of the larger car class because of their smaller size, this proved the method's ability to represent the shape of the objects differently depending on the distance. Figure 14 shows the resulting images of the synthetic LiDAR point cloud generation for the pedestrian and cyclist classes.

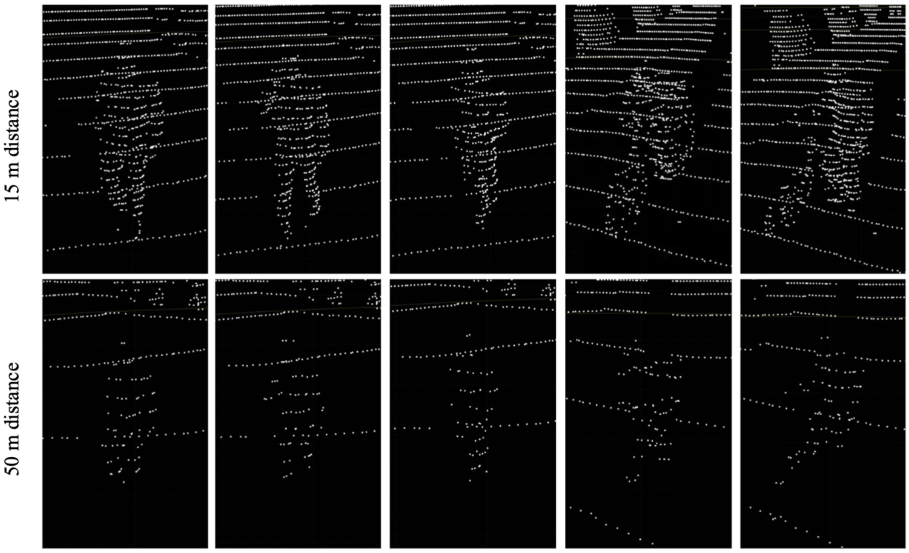


Figure 14. Experimental results of synthetic LiDAR point cloud generation (pedestrian and cyclist classes).

4.2.3. Platooning

Figure 9 shows the platooning situation that appears in the real KITTI 360 dataset, and the other image shows a situation in which the synthetic object points generated through the proposed framework depict platooning. In the figure, two synthetic object points are generated, both of which follow the direction of nearby real objects by following two steps: (1) calculating the pose decision area based yaw value determination and (2) updating the pose of the synthetic object to point toward a nearby real vehicle's pose, as stated in Section 3.1.3. In summary, the yaw angle is assigned based on the pose decision area, and then it can be updated based on the orientation of nearby real vehicles. If a real vehicle is within a certain proximity to the synthetic object, the yaw value of the virtual object is updated to match the yaw angle of the nearest real vehicle, allowing for a platooning process, as shown in Figure 15f. However, the current study should further explore complex platooning mechanisms besides the angle alone to implement better pose estimation techniques. This aspect is considered as a potential future scope in terms of platooning scenarios.

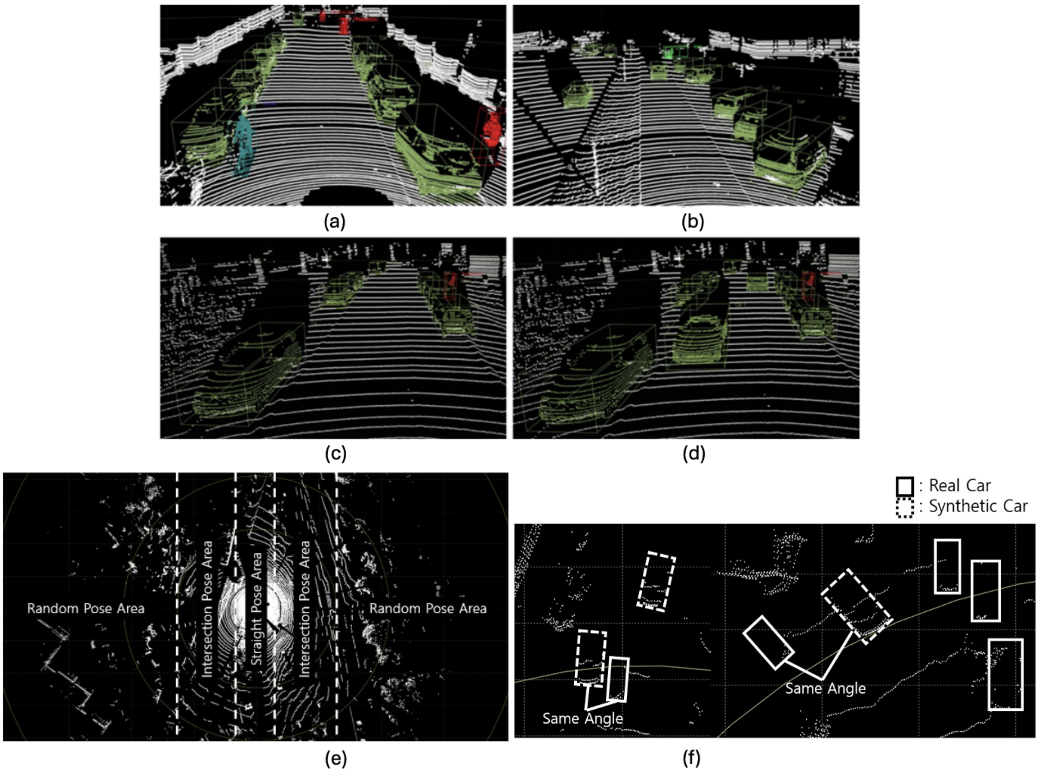


Figure 15. Platooning represented by proposed method where green represents car objects, light green represents van objects, blue represents cyclist objects, and red represents pedestrian objects. (**a**,**b**) Frame presenting the platooning situation included in the KITTI 360 dataset; (**c**) original input LiDAR scene; (**d**) output scene with 2 synthetic car objects; (**e**) pose decision areas for platooning with respect to a real vehicle; (**f**) pose of a synthetic object determined by a nearby real vehicle for platooning.

4.3. Deep Learning Model Training

The KITTI 360 dataset is a large-scale autonomous driving dataset consisting of approximately 15,000 frames of data in the form of a fusion of 2D image data and 3D LiDAR sensor data. Approximately 15,000 frames of data are provided in the form of Train, Val, and Test sets in a ratio of 2:1:1. Most models in the 3D object detection field use the KITTI 360 dataset to demonstrate object recognition performance, and a benchmark suite is provided for this purpose so that developers of object recognition deep learning models can access and utilize it. To confirm that the synthetic object points generated through the proposed framework can be effectively used for deep learning model training on RAM 8 GB Intel i5/GTX 1080 8 GB, datasets augmented with synthetic object points were used to train various deep learning models and evaluate their performance. The deep learning model training experiments were conducted separately using the datasets augmented with the car class and those augmented with the pedestrian and cyclist classes.

4.3.1. Car Class

A dataset augmentation experiment using synthetic object points from the car class was conducted by training various 3D object detection models on the augmented dataset and measuring the improvement in training performance. The KITTI 360 dataset was used, and the PointPillars, PV-RCNN, and Voxel R-CNN models were used as the deep learning models. The performance of the proposed framework was evaluated in comparison with an existing research method, LiDAR-Aug. Table 5 lists the car class performances of the proposed framework. Since Voxel-RCNN was published before LiDAR-Aug, this item is vacant, and the training performance of the dataset augmented with the proposed framework improved compared with the previous ones for all three selected models. The extent of improvement in training performance showed a trend in which it was higher when the original performance was lower and somewhat lower when the original performance was higher. It was also observed that the performance of the proposed framework improved compared to that of LiDAR-Aug for all the models. The performance of LiDAR-Aug for the voxel R-CNN model was not included as it was not presented in this paper.

Table 5. Evaluation results of proposed framework for the car class on the KITTI validation dataset.

Methods	**AP3D (%)**			**mAP**
	Easy	**Moderate**	**Hard**	
PointPillars with KITTI	85.41	73.59	68.76	75.92
PointPillars with LiDAR-Aug	87.75	77.83	74.90	80.16
PointPillars with proposed work (Ours)	88.75	80.43	77.52	82.23
PV-RCNN with KITTI	88.86	78.83	78.30	82.00
PV-RCNN with LiDAR-Aug	90.18	84.23	78.95	84.45
PV-RCNN with proposed work (Ours)	92.64	84.54	82.98	86.72
Voxel R-CNN with KITTI	92.24	85.01	82.51	86.59
Voxel R-CNN with LiDAR-Aug	NB	NB	NB	NB
Voxel R-CNN with Proposed (Ours)	92.67	85.34	83.13	87.05

4.3.2. Pedestrian and Cyclist Classes

The dataset augmentation experiment for the pedestrian and cyclist classes was conducted in a similar manner using the augmented KITTI 360 dataset to train the PointPillar and PV-RCNN models and to calculate the improvement in training performance. The performance of the dataset augmentation carried out via the proposed framework was also validated by comparing it with the performance of LiDAR-Aug. Tables 6 and 7 list the performance of the proposed framework for the pedestrian and cyclist classes. The performance of the proposed framework for the pedestrian and cyclist classes was slightly improved compared with the original dataset. However, when compared to the LiDAR-Aug method, the proposed framework did not outperform LiDAR-Aug in every case. The performance of the cyclist class was not included as it was not presented in the LiDAR-Aug paper.

Interestingly, although the dataset was augmented for the pedestrian and cyclist classes, the training performance for the car class improved. This is interpreted as dataset augmentation mitigating the object imbalance between classes that existed because the original dataset had fewer pedestrian and cyclist class objects than car class objects, thereby enhancing the learning performance for the car class as well.

Table 6. Evaluation results of proposed work for the pedestrian class on KITTI validation dataset using pointpillars.

Methods	AP3D (%)								
	Car			Pedestrian			Cyclist		
	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
PointPillars with KITTI	85.41	73.59	68.76	47.51	43.82	42.20	84.64	64.26	60.69
PointPillars with LiDAR-Aug	87.75	77.83	74.90	59.99	55.15	52.66	-	-	-
PointPillars with Proposed work	87.99	78.42	75.35	56.28	50.5	46.07	84.68	64.63	61.12

Table 7. Evaluation results of proposed framework for the pedestrian class on KITTI validation dataset using PV-RCNN.

Methods	AP3D (%)								
	Car			Pedestrian			Cyclist		
	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
PV-RCNN with KITTI	88.86	78.83	78.30	60.56	53.75	51.90	90.24	71.83	68.33
PV-RCNN with LiDAR-Aug	90.18	84.23	78.95	65.05	58.90	55.52	-	-	-
PV-RCNN with Proposed (Ours)	92.00	84.68	82.67	66.41	59.45	53.62	90.85	72.42	69.04

5. Limitations and Future Work

The proposed framework consists of several limitations, such as the fact that the performed experiments primarily focused on evaluating metrics such as accuracy (mAP) in favor of the training process. Dataset diversification must be carried out by introducing vast datasets for LiDAR and employing the proposed framework to improve training performance. However, including a variety of metrics such as the real-to-synthetic noise distribution and testing for realism would favor the overall integration of the framework into the training process. Additionally, other evaluation models could be used aside from those included in the current study, which were limited to PointPillar, PV R-CNN, and Voxel R-CNN. Due to the lack of open-source SOTA resources for reproducibility, this study employed the KITTI dataset to perform the evaluation, alongside LiDAR-Aug. Subsequent research may extend this process to produce integrated 2D and 3D virtual entities with calibrated LiDAR and picture data, thereby reducing the time and resources necessary for synthetic LiDAR point cloud generation. Also, this study did not consider return losses according to the reflected intensity, which requires further investigation regarding materials and reflectivity. The idea of reflectance was not explored in this study, which is a shortcoming, and it should be explored in future work. Additionally, various complex platooning mechanisms must be explored with better pose estimation variables besides the angle.

6. Conclusions

The proposed framework includes a module to determine the pose of synthetic object points, along with an automated system capable of representing both near and distant synthetic object points, as well as platooning scenarios for vehicles on the road. The proposed framework was assessed by qualitative and quantitative performance analyses on the synthesis of objects within an established dataset, namely KITTI. The integration of the synthetic object through this framework in the augmented dataset demonstrated that synthetic object points can be efficiently utilized in training deep learning models for 3D object detection applications. This study showed that the proposed framework can accurately represent distant objects and produce synthetic object points that closely align with real-world distributions, in contrast to the existing LiDAR-Aug technique. The performance of the proposed framework was evaluated on various 3D detection models, such as PointPillars, PV-RCNN, and Voxel R-CNN, for the KITTI dataset. The results indicated an increase in mAP (mean average precision) by 1.97%, 1.3%, and 0.46% from the original dataset values of 82.23%, 86.72%, and 87.05%, respectively. The proposed method has to deal with return loss in new projected points, which is a shortcoming at this stage. Future investigations may expand this methodology to generate integrated 2D and 3D virtual entities with calibrated LiDAR and image data, thereby minimizing the time and resources required for AI dataset generation and enhancing autonomous driving technology.

Author Contributions: Conceptualization, S.L., K.K. and H.K.; methodology, S.L. and K.K.; software, S.L. and K.K.; validation, S.L., V.K., K.K. and H.K.; formal analysis, S.L. and K.K.; investigation, K.K.; resources, V.K. and H.K.; data curation, K.K., X.L. and V.K; writing—original draft preparation, S.L. and K.K.; writing—review and editing, V.K.; visualization, X.L. and K.K.; supervision, H.K.; project administration, K.K.; funding acquisition, H.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the BK21 Four Program, funded by the Ministry of Education (MOE, Korea) and the National Research Foundation of Korea (NRF).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data underlying the conclusions of this article will be made available by the corresponding author upon reasonable request.

Conflicts of Interest: Author Sangjun Lee was employed by the company Hyundai KEFICO Corp. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Esmorís, A.M.; Weiser, H.; Winiwarter, L.; Cabaleiro, J.C.; Höfle, B. Deep learning with simulated laser scanning data for 3D point cloud classification. *ISPRS J. Photogramm. Remote Sens.* **2024**, *215*, 192–213. [CrossRef]
2. Beltrán, J.; Cortés, I.; Barrera, A.; Urdiales, J.; Guindel, C.; García, F.; de la Escalera, A. A method for synthetic LiDAR generation to create annotated datasets for autonomous vehicles perception. In Proceedings of the 2019 IEEE Intelligent Transportation Systems Conference (ITSC), Auckland, New Zealand, 27–30 October 2019; pp. 1091–1096.
3. Yue, X.; Wu, B.; Seshia, S.A.; Keutzer, K.; Sangiovanni-Vincentelli, A.L. A lidar point cloud generator: From a virtual world to autonomous driving. In Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval, Yokohama, Japan, 11–14 June 2018; pp. 458–464.
4. Wang, F.; Zhuang, Y.; Gu, H.; Hu, H. Automatic generation of synthetic LiDAR point clouds for 3-D data analysis. *IEEE Trans. Instrum. Meas.* **2019**, *68*, 2671–2673. [CrossRef]
5. Hossny, M.; Saleh, K.; Attia, M.; Abobakr, A.; Iskander, J. Fast synthetic LiDAR rendering via spherical UV unwrapping of equirectangular Z-buffer images. In Proceedings of the Computer Vision and Pattern Recognition, Image and Video Processing, Glasgow, UK, 23–28 August 2020.
6. Chitnis, S.A.; Huang, Z.; Khoshelham, K. Generating Synthetic 3D Point Segments for Improved Classification of Mobile LIDAR Point Clouds. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2021**, *43*, 139–144. [CrossRef]
7. Yin, T.; Gastellu-Etchegorry, J.P.; Grau, E.; Lauret, N.; Rubio, J. Simulating satellite waveform Lidar with DART model. In Proceedings of the 2013 IEEE International Geoscience and Remote Sensing Symposium-IGARSS, Melbourne, Australia, 21–26 July 2013; pp. 3029–3032.
8. Yin, T.; Lauret, N.; Gastellu-Etchegorry, J.P. Simulation of satellite, airborne and terrestrial LiDAR with DART (II): ALS and TLS multi-pulse acquisitions, photon counting, and solar noise. *Remote Sens. Environ.* **2016**, *184*, 454–468. [CrossRef]
9. Gastellu-Etchegorry, J.P.; Yin, T.; Lauret, N.; Cajgfinger, T.; Gregoire, T.; Grau, E.; Feret, J.B.; Lopes, M.; Guilleux, J.; Dedieu, G.; et al. Discrete anisotropic radiative transfer (DART 5) for modeling airborne and satellite spectroradiometer and LIDAR acquisitions of natural and urban landscapes. *Remote Sens.* **2015**, *7*, 1667–1701. [CrossRef]
10. Gastellu-Etchegorry, J.P.; Yin, T.; Lauret, N.; Grau, E.; Rubio, J.; Cook, B.D.; Morton, D.C.; Sun, G. Simulation of satellite, airborne and terrestrial LiDAR with DART (I): Waveform simulation with quasi-Monte Carlo ray tracing. *Remote Sens. Environ.* **2016**, *184*, 418–435. [CrossRef]
11. Yang, X.; Wang, Y.; Yin, T.; Wang, C.; Lauret, N.; Regaieg, O.; Xi, X.; Gastellu-Etchegorry, J.P. Comprehensive LiDAR simulation with efficient physically-based DART-Lux model (I): Theory, novelty, and consistency validation. *Remote Sens. Environ.* **2022**, *272*, 112952. [CrossRef]
12. Yang, X.; Wang, C.; Yin, T.; Wang, Y.; Li, D.; Lauret, N.; Xi, X.; Wang, H.; Wang, R.; Wang, Y.; et al. Comprehensive LiDAR simulation with efficient physically-based DART-Lux model (II): Validation with GEDI and ICESat-2 measurements at natural and urban landscapes. *Remote Sens. Environ.* **2025**, *317*, 114519. [CrossRef]

13. Fang, J.; Zuo, X.; Zhou, D.; Jin, S.; Wang, S.; Zhang, L. Lidar-aug: A general rendering-based augmentation framework for 3D object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 4710–4720.
14. Xiao, A.; Huang, J.; Guan, D.; Zhan, F.; Lu, S. Transfer learning from synthetic to real lidar point cloud for semantic segmentation. In Proceedings of the AAAI Conference on Artificial Intelligence, Online, 22 February–1 March 2022; Volume 36, pp. 2795–2803.
15. Xiang, Z.; Huang, Z.; Khoshelham, K. Synthetic lidar point cloud generation using deep generative models for improved driving scene object recognition. *Image Vis. Comput.* **2024**, *150*, 105207. [CrossRef]
16. Zhao, J.; Zheng, P.; Ma, R. D-Aug: Enhancing Data Augmentation for Dynamic LiDAR Scenes. *arXiv* **2024**, arXiv:2404.11127.
17. Zhang, Y.; Ding, M.; Yang, H.; Niu, Y.; Ge, M.; Ohtani, K.; Zhang, C.; Takeda, K. LiDAR Point Cloud Augmentation for Adverse Conditions Using Conditional Generative Model. *Remote Sens.* **2024**, *16*, 2247. [CrossRef]
18. Park, J.; Kim, K.; Shim, H. Rethinking Data Augmentation for Robust LiDAR Semantic Segmentation in Adverse Weather. *arXiv* **2024**, arXiv:2407.02286.
19. Reichardt, L.; Uhr, L.; Wasenmüller, O. Text3DAug–Prompted Instance Augmentation for LiDAR Perception. *arXiv* **2024**, arXiv:2408.14253.
20. López, A.; Ogayar, C.J.; Jurado, J.M.; Feito, F.R. A GPU-accelerated framework for simulating LiDAR scanning. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 1–18. [CrossRef]
21. Winiwarter, L.; Pena, A.M.E.; Weiser, H.; Anders, K.; Sánchez, J.M.; Searle, M.; Höfle, B. Virtual laser scanning with HELIOS++: A novel take on ray tracing-based simulation of topographic full-waveform 3D laser scanning. *Remote Sens. Environ.* **2022**, *269*, 112772. [CrossRef]
22. Anand, V.; Lohani, B.; Pandey, G.; Mishra, R. Toward Physics-Aware Deep Learning Architectures for LiDAR Intensity Simulation. *arXiv* **2024**, arXiv:2404.15774.
23. Zyrianov, V.; Che, H.; Liu, Z.; Wang, S. LidarDM: Generative LiDAR Simulation in a Generated World. *arXiv* **2024**, arXiv:2404.02903.
24. Eggert, M.; Schade, M.; Bröhl, F.; Moriz, A. Generating Synthetic LiDAR Point Cloud Data for Object Detection Using the Unreal Game Engine. In Proceedings of the International Conference on Design Science Research in Information Systems and Technology, Trollhättan, Sweden, 3–5 June 2024; Springer: Berlin/Heidelberg, Germany, 2024; pp. 295–309.
25. Manivasagam, S.; Wang, S.; Wong, K.; Zeng, W.; Sazanovich, M.; Tan, S.; Yang, B.; Ma, W.C.; Urtasun, R. Lidarsim: Realistic lidar simulation by leveraging the real world. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11167–11176.
26. Li, R.; Li, X.; Heng, P.A.; Fu, C.W. Pointaugment: An auto-augmentation framework for point cloud classification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 6378–6387.
27. Lee, S.; Lim, H.; Myung, H. Patchwork++: Fast and robust ground segmentation solving partial under-segmentation using 3D point cloud. In Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, 23–27 October 2022; pp. 13276–13283.
28. Geiger, A.; Lenz, P.; Urtasun, R. Are we ready for autonomous driving? The kitti vision benchmark suite. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 3354–3361.
29. Caesar, H.; Bankiti, V.; Lang, A.H.; Vora, S.; Liong, V.E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; Beijbom, O. Nuscenes: A multimodal dataset for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11621–11631.
30. Lang, A.H.; Vora, S.; Caesar, H.; Zhou, L.; Yang, J.; Beijbom, O. Pointpillars: Fast encoders for object detection from point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 12697–12705.
31. Shi, S.; Guo, C.; Jiang, L.; Wang, Z.; Shi, J.; Wang, X.; Li, H. Pv-rcnn: Point-voxel feature set abstraction for 3D object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 10529–10538.
32. Deng, J.; Shi, S.; Li, P.; Zhou, W.; Zhang, Y.; Li, H. Voxel r-cnn: Towards high performance voxel-based 3D object detection. In Proceedings of the AAAI Conference on Artificial Intelligence, Online, 2–9 February 2021; Volume 35, pp. 1201–1209.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

sensors

Article

PointCloud-At: Point Cloud Convolutional Neural Networks with Attention for 3D Data Processing

Saidu Umar and Aboozar Taherkhani *

School of Computer Science and Informatics, De Montfort University, Leicester LE1 9BH, UK; saeedumar5@gmail.com
* Correspondence: aboozar.taherkhani@dmu.ac.uk

Abstract: The rapid growth in technologies for 3D sensors has made point cloud data increasingly available in different applications such as autonomous driving, robotics, and virtual and augmented reality. This raises a growing need for deep learning methods to process the data. Point clouds are difficult to be used directly as inputs in several deep learning techniques. The difficulty is raised by the unstructured and unordered nature of the point cloud data. So, machine learning models built for images or videos cannot be used directly on point cloud data. Although the research in the field of point clouds has gained high attention and different methods have been developed over the decade, very few research works directly with point cloud data, and most of them convert the point cloud data into 2D images or voxels by performing some pre-processing that causes information loss. Methods that directly work on point clouds are in the early stage and this affects the performance and accuracy of the models. Advanced techniques in classical convolutional neural networks, such as the attention mechanism, need to be transferred to the methods directly working with point clouds. In this research, an attention mechanism is proposed to be added to deep convolutional neural networks that process point clouds directly. The attention module was proposed based on specific pooling operations which are designed to be applied directly to point clouds to extract vital information from the point clouds. Segmentation of the ShapeNet dataset was performed to evaluate the method. The mean intersection over union (mIoU) score of the proposed framework was increased after applying the attention method compared to a base state-of-the-art framework that does not have the attention mechanism.

Keywords: deep learning; point cloud data; attention mechanism; 3D data

1. Introduction

Point cloud processing using deep learning methods has gained a lot of attention. Point clouds are a set of data points in space to display 3D geometry. They have gained popularity and wide usage in several domains. The rapid growth in 3D technologies and 3D sensors has made point cloud data increasingly available [1]. Additionally, the world is in three dimensions, thus point clouds are a suitable format for representing the real world in XYZ coordinates. The usage of point clouds extends to a variety of disciplines and the 3D nature of point clouds makes them the appropriate format for autonomous driving, robotics, virtual and augmented reality, heritage preservation, and many more applications [2]. The availability of point cloud data has raised a need for advanced deep learning methods to process 3D point clouds.

Deep learning techniques have been used to perform different kinds of processing on image and video data [3–6]. Some deep learning techniques that have been applied to point clouds include classification, detection and tracking, reconstruction, and segmentation [7]. For over 50 years, the segmentation of images has been the focus of several researchers. The purpose of segmentation is to break an image into subregions with similar features, and it has had a great impact on computer vision [7,8]. Point cloud segmentation has faced challenges due to the unstructured nature of the point cloud data and its highly redundant and uneven nature [9]. Although various point cloud segmentation methods exist, designing deep learning techniques for this purpose remains a challenging task. Developing advanced methods in this field will improve the accuracy of computer vision in 3D space for different applications.

Attention mechanisms have had a great impact on deep learning by improving the accuracy of models and their performance [10]. Attention mechanisms help the models concentrate on the most important features of input data [11]. There are several types of attention mechanisms. One of the early applications of attention mechanisms in point cloud data is the shuffle attention model [12]. A little research has been carried out on improving the performance of point cloud convolutional neural networks (CNNs) with attention mechanisms. Unlike pixel images, there have been few usages of attention mechanisms directly applied to point cloud data [7]. The unstructured and unordered nature of point cloud data causes each point to have specific importance, and using a processing approach that evenly processes the points is not suitable.

Therefore, in this research, an attention mechanism is proposed for processing point cloud data. The attention mechanism is directly applied to point cloud data without mapping them to a continuous space. It learns the importance of points using training data and places specific emphasis on each point. The proposed attention mechanism is added to a network called ConvPoint [9]. The performance of the proposed method was compared with the base method without the attention mechanism, i.e., ConvPoint [9]. Additionally, the method was compared with other state-of-the-art methods.

The structure of this paper is as follows. Section 2 reviews the existing literature on deep learning methods for point clouds and attention mechanisms. Section 3 explores the proposed method and its components. Section 4 discusses experiments and results. Finally, Section 5 concludes the paper.

2. Literature Review

For different applications, such as perception and localisation, which are key for navigation in autonomous vehicles, visual data processing plays an important role [13]. Image data extracted from a camera are usually represented in 2D; however, the 2D data lack the required geometric and volumetric information [13]. A point cloud is a representation tool for 3D data. In this section, initially, deep learning methods for point clouds are reviewed. Then, attention mechanisms in deep learning for images and point clouds are discussed.

2.1. Deep Learning Methods for Point Clouds

A key challenge in point cloud processing is the inefficiency of traditional CNNs in processing the original form of point cloud data [14]. The sparse, unstructured, and unordered nature of point clouds makes the standard CNN architecture less effective. To address these issues, refs. [15,16] proposed the pre-processing of point cloud data into voxel representations or 2D images. Three-dimensional point clouds can be converted into 2D images using multi-view-based methods or into a 3D volumetric representation to be processed by well-known 2D or 3D convolutional networks [7]. Although these methods might ease the implementation, the loss of certain geometric information of the point cloud data is the cost of this implementation. This significant gap represents a critical challenge in the field and identifies a huge demand for approaches that can directly process point cloud data without compromising its intrinsic characteristics. Our proposed method, which is a point-based approach, addresses this limitation by leveraging the power of deep learning to develop an architecture that preserves the original structure of the point cloud data, thus maintaining important geometric and spatial information. The pointwise method does not use any voxelization or other projection methods.

There are different types of pointwise methods. Pointwise MLP methods process each point independently using shared multilayer perceptrons (MLPs). Then, the output features are aggregated (global aggregation) using an aggregation function. PointNet [17] is a pioneering pointwise MLP method that uses max pooling for global feature aggregation. It was improved by introducing a structure to extract global and local features to create PointNet++ [14]. PointNet++ uses a max pooling method for local feature aggregation. The dynamic graph CNN (DGCNN) [15] was proposed to aggregate local region information using the feature of a centre point and the differences between the feature of the centre point and the features of its k nearest neighbour points. DGCNN only considers the pair relation for the centre point. PointWeb [16], a pointwise MLP method, was proposed to consider all pairs of points in a local region using a module called adaptive feature adjustment (AFA). One of the challenges of the pointwise MLP methods is their high computational cost.

Convolution-based methods for 3D point clouds are another group of deep neural networks that use specific convolution kernels to process 3D point clouds. Three-dimensional discrete convolution methods are an important group of convolution-based methods, and they use convolutional kernels on regular grids and the offset of each point to a centre point to define weights for neighbouring points. For instance, Hua et al. [18] used uniform grids to define convolutional kernels. They transformed 3D point clouds into uniform grids and assigned the same weight to the points falling in the same cell or subdomain. The mean value of the features of the points in a cell is multiplied by its corresponding kernel weight and summed with the other weighted mean values on all other cells in the kernel domain to calculate the output. In the spherical convolutional kernel proposed by Lei et al. [19], multiple volumetric bins were created by partitioning a 3D spherical neighbouring region. A learnable weighting matrix was assigned for each bin.

Three-dimensional continuous convolution methods are another group of convolution-based methods for processing 3D point clouds. Despite the 3D discrete convolution methods that consider discrete regions or domains, they used convolutional kernels on a continuous space. The weights in the convolutional kernel for neighbouring points are related to the continuous spatial distance from a centre point. For instance, RS-Conv [20] uses an MLP to implement a convolution. The MLP is trained to map low-level relations between input points such as Euclidean distance and the relative position to high-level relations between points in the local subset. Then, the output of the MLP is used to calculate the weighted sum over the given subset. ConvPoint [9] is another method that performs convolution in two parts, namely the spatial and feature parts. The locations of the kernel points are selected randomly from a unit sphere. The kernel points and the position of input points are applied to an MLP to create kernel weights. The convolutional layer can be used as a building block of complex networks. Different structures in the classical neural networks can be used to design new network structures for point cloud processing. In this project, an attention mechanism is used with a 3D continuous convolution method to design a new network structure.

Wang et al. [8] addressed one of the critical limitations, the inability of discrete CNNs to handle the unstructured nature of point cloud data, thus proposing a framework that generalises discrete CNNs to deal with point clouds. While [8] made progress in adapting CNNs to process point cloud data, the framework faces challenges in handling large-scale, real-time point cloud data, which is critical for many practical applications. The performance of this framework in scenarios with varying point densities or occlusions remains unclear. Our work is expected to build upon this framework to reduce some of the limitations and to improve point cloud processing in real-world applications and dynamic environments.

2.2. Attention Mechanisms in Deep Learning Methods for Images

Attention mechanisms have improved image segmentation significantly; however, their applications in point cloud data have been limited. There have been promising applications of attention mechanisms in 2D image processing. Approaches applied to 2D images include spatial channel attention [10], shuffle attention [11], and the convolution block attention module [21]. These approaches have improved model performance by effectively capturing channel dependencies and pixel-level relationships.

Using image-based attention mechanisms directly on 3D point cloud data has been faced with challenges. Often, preprocessing or data conversion is required, leading to potential data loss. Additionally, the unique spatial structure of the 3D point cloud data may not be fully leveraged.

Our work aims to bridge this gap by developing attention mechanisms designed to work on 3D point cloud data. This framework will address the challenges caused by the unstructured nature of point cloud data while maintaining the advantages of attention mechanisms.

2.3. Attention Mechanisms in Deep Learning for Point Clouds

Yang et al. [22] proposed the attention-based point network (AttPNet), a network that utilises attention mechanisms to perform channel weighting and global feature masking on feature areas. AttPNet has two branches, where one branch deduces global features from point sets using convolutional layers to create a channel attention block focusing on the key channels of the data. The other branch performs the calculation of an attention mask for every point. Subsequently, the authors designed a point cloud dataset of electron cryo-tomography (ECT) and used these data to show the AttPnet's capacity of handling fine-grained structures. The authors aimed to design a model that handles fine-grained structures. The attention mechanisms use the MLP. Additionally, they only use the features, and the exact position of the points was not applied to the MLP. In our research, a point cloud convolutional layer that accepts the position of the input points in addition to the input features is used to design an attention mechanism.

Hu et al. [23] introduced an attention-based module for extracting local features in their semantic framework for point cloud data labelling. Although this design achieved a modest output, it had limitations in fully utilising geometric calculations of neighbouring points. Deng and Dong [24] designed a global attention network for point cloud segmentation to address the problem of learning long-range dependencies from 3D point clouds, which has been a challenging problem in the processing of 3D point clouds. The global attention network, or GA-Net, comprises a global attention module that is point-independent and another global attention module that is point-dependent for gathering background information on 3D points. Both [25,26] made significant progress in utilising geometric calculations of neighbouring points and learning long-range dependencies but face limitations in balancing computational complexity with performance.

Several researchers focus on spatial encodings whilst ignoring the channel relationships, making feature learning insufficient. Hence, the lightweight attention module (LAM) was developed in [27] to improve the performance while adopting a new convolutional function and introducing a channel-based attention mechanism. However, its integration with existing networks may not fully exploit the unique properties of point cloud data.

One of the issues with working with point clouds is the inability to fully utilise the geometric information of neighbouring points. This prompted Feng et al. [28] to propose and design the local attention–edge convolution (LEA-Conv) layer. This layer is an extension of the works proposed in [14,15,29]. The LAE-Conv model builds a graph of neighbourhood points along several routes. Consequently, a search strategy was proposed in [28] to use a multidirectional search to find all points in the neighbourhood across 16 directions systematically within a ball query to generalise the local geometric shape over the space. Additionally, a pointwise spatial attention block was proposed to capture information in the spatial dimension. The output features of the LEA-Conv layer were applied to the spatial attention block to create outputs that capture the spatial dependency of the points. The spatial attention block does not consider the correlation in the different channels. In this research, a channel attention mechanism is proposed.

This review highlighted the importance of attention mechanisms on model performance. A critical gap exists in designing attention mechanisms that work directly on the geometric information contained in 3D point cloud data. These attention mechanisms will significantly improve the accuracy of existing 3D point CNNs.

3. Methodology and Framework

This section covers a brief description of ConvPoint [9], which is used as the base of the proposed method after a discussion of the problem statement. The point cloud convolutional layer in [9] is used as a base method in this research because it is a convolution method that directly processes point clouds. It does not have an attention mechanism. Moreover, its output is also point clouds. Therefore, its output has an acceptable format that can be applied to the proposed attention mechanism in this paper, which is designed to directly work on point cloud data. The proposed attention mechanism will be introduced in Section 3.3. Lastly, the final network structure will be discussed.

3.1. Problem Statement

In this research, a method is proposed to enhance the performance of ConvPoint [9] with a spatial attention module that is inspired by the convolutional block attention module (CBAM) [21]. ConvPoint is an oversimplification of a discrete convolutional neural network [9]. The CBAM is an attention module for the usual convolutional neural networks used for normal data such as images [21]. The structure of point cloud data is different from the usual data produced by a classical convolutional layer and consequently, it needs a new attention mechanism. In this research, we design a channel attention mechanism to boost the performance of ConvPoint [9].

Suppose a convolutional layer works directly on point clouds. The layer has a kernel function and an input in the form of point clouds. In the convolutional layer for point clouds, the following kernel K and input P that have compatible dimensionality are used: $K = \{(\boldsymbol{c}_i, \boldsymbol{w}_i), \boldsymbol{c}_i \in \mathbb{R}^3, \boldsymbol{w}_i \in \mathbb{R}^{\boldsymbol{n}}, i \in [\![1, |K|]\!]\}$ and $P = \{(\boldsymbol{p}, \boldsymbol{x})\} = \{(\boldsymbol{p}_i, \boldsymbol{x}_{\boldsymbol{i}}), \boldsymbol{p}_i \in \mathbb{R}^3, \boldsymbol{x}_i \in \mathbb{R}^{\boldsymbol{n}}, \boldsymbol{i} \in [\![1, |P|]\!]\}$, where $|K|$ is the number of elements in the kernel and $|P|$ shows the size of the input set, i.e., the number of points that are in the input set P. The convolutional layer for point clouds accepts point cloud data composed of several input points and corresponding features, i.e., $P = \{(\boldsymbol{p}, \boldsymbol{x})\}$, where $\boldsymbol{x}$ is an n-dimensional feature from the input feature space corresponding to an input point, i.e., $\boldsymbol{p}$ in the 3D space. The convolutional layer uses the kernel K to create an output composed of the place of the output points, i.e., $\boldsymbol{q}$, and their features ($\boldsymbol{y}$), i.e., $Q = \{(\boldsymbol{q}, \boldsymbol{y})\} = \{(\boldsymbol{q}_i, \boldsymbol{y}_{\boldsymbol{i}}), \boldsymbol{q}_i \in \mathbb{R}^3, \boldsymbol{y}_i \in \mathbb{R}^{\boldsymbol{n}}, \boldsymbol{i} \in [\![1, |Q|]\!]\}$.

3.2. Continuous Convolutions for Point Cloud Processing (ConvPoint)

A continuous convolutional layer was proposed in [9] by adjusting the discrete convolutional layer used for the common 2D image datasets to process point cloud data. The following two operations are performed in ConvPoint [9] to map the input P to the output Q: 1—point selection, and 2—convolution on point sets.

(a) Point selection.

Each point $\boldsymbol{q}$ in the output set, $Q = \{(\boldsymbol{q}, \boldsymbol{y})\}$, is selected randomly from input points that are in $P = \{(\boldsymbol{p}, \boldsymbol{x})\}$ using the random method in [9]. A score is allocated to each input point and whenever a point is selected randomly, its score is increased by 100. Additionally, the scores of its neighbour points are increased by 1. Increasing the scores of a selected point and its neighbour points reduces their chance of being selected in the next selection procedure. This method is used to give a chance to all points to be selected and to reduce the probability of selecting repeated points. The selection procedure is continued until the required output points are selected.

(b) Convolution on point sets.

After selecting the output points, for each output point $\boldsymbol{q}$, the *k-d* tree method [9] is used to find local neighbour points in $P = \{(\boldsymbol{p}, \boldsymbol{x})\}$ to create a subset of points for each $\boldsymbol{q}$. Then, point convolution is applied to the subset of points using (1).

$$y = \beta + \frac{1}{|P|} \sum_{j=1}^{|P|} \sum_{i=1}^{|k|} \boldsymbol{w}_i \boldsymbol{x}_j f\left(\boldsymbol{p}_j - \boldsymbol{c}\right) \tag{1}$$

where β is a bias parameter, $\frac{1}{|P|}$ is used to reflect the input set size to have robustness against input size variation, and $f(.)$ is a geometrical weighting function to distribute the input $P = \{(\boldsymbol{p}, \boldsymbol{x})\}$ onto the kernel $K = \{(\boldsymbol{c}, \boldsymbol{w})\}$. It accepts the relative distance between each input point $\boldsymbol{p}$ and all the kernel elements $\{\boldsymbol{c}\}$, i.e., $\boldsymbol{p}_j - \boldsymbol{c}$, to create a weight in $\mathbb{R}$ corresponding to an input point as shown in (2).

$$f : \mathbb{R}^3 \times \left(\mathbb{R}^3\right)^{|K|} \rightarrow \mathbb{R} \tag{2}$$

A simple neural network, i.e., the multilayer perceptron (MLP), is trained to act as $f(.)$. The MLP is used to build the general function $f(.)$, as this approach is easier than building the general function from scratch. For each kernel in the convolution operation, spatial and feature parts were processed independently. The parameters were optimised using gradient descent during training.

The locations of the kernel elements $\{\boldsymbol{c}\}$ are initialised by randomly selecting them from the unit sphere. Training parameters of the convolutional layer, i.e., $\{\boldsymbol{c}\}$ and $\{\boldsymbol{w}\}$, and the training parameters of the MLP are optimised using the gradient descent method.

3.3. The Proposed Attention Mechanism for Point Cloud Continuous Convolutional Layer

In this research, the attention module proposed in [21] for the common convolutional layer inspired us to design an attention mechanism for the point cloud convolutional neural network described in Section 3.2. A channel attention mechanism for the continuous convolutional layer for point clouds (ConvPoint) is proposed in this research. The proposed attention mechanism is designed to perform different operations such as average pooling and max pooling operations on point cloud data extracted from a ConvPoint layer.

In the proposed attention mechanism, an input feature map is converted into a channel attention map using the proposed method. The process of channel attention is given by (3).

$$\boldsymbol{F}' = \boldsymbol{M_c}(P) \bigotimes P \tag{3}$$

where $P = \{(\boldsymbol{p}, \boldsymbol{x})\}$ is the input to the attention block, which is produced from its previous ConvPoint layer, $\otimes$ denotes multiplication of the elements, and $\boldsymbol{M_c}(.)$ is a channel attention mechanism. As the input of ConvPoint is composed of two parts, i.e., $\boldsymbol{p}$ and $\boldsymbol{x}$, in this paper, a mechanism was proposed to deal with the two parts of the attention mechanism.

First, pooling operations, i.e., max pooling and average pooling, were applied to the features $X = \left[\boldsymbol{x}_1 \boldsymbol{x}_2 \ldots \boldsymbol{x}_{|p|}\right]$, where $\boldsymbol{x}_i \in \mathbb{R}^{\boldsymbol{n}}$ for $\boldsymbol{i} \in [\![1, |P|]\!]$, $X_{max} \in \mathbb{R}^{\boldsymbol{n}}$, and $X_{mean} \in \mathbb{R}^{\boldsymbol{n}}$ are created after the max pooling and average pooling on the features related to the $|P|$ input points. Then, the results are concatenated as shown in Figure 1 using (4).

$$X^c = C(X_{max}, X_{mean}) \tag{4}$$

where $C(.)$ is the concatenation function combining the two inputs with a size of $(1, n)$ to create a concatenated output with a size of $(2, n)$.

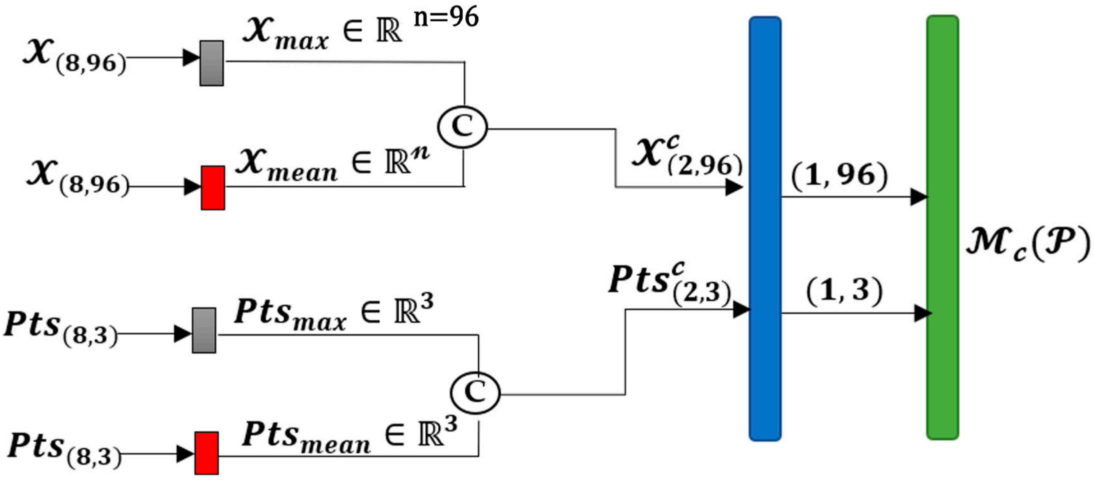

$\mathcal{X}_{(8,96)}$
$\mathcal{X}_{max} \in \mathbb{R}^{n=96}$
$\mathcal{X}_{(8,96)}$
$\mathcal{X}_{mean} \in \mathbb{R}^{n}$
C
$\mathcal{X}^{c}_{(2,96)}$
(1, 96)
$\mathcal{M}_c(\mathcal{P})$
$Pts_{(8,3)}$
$Pts_{max} \in \mathbb{R}^3$
$Pts^{c}_{(2,3)}$
(1, 3)
C
$Pts_{(8,3)}$
$Pts_{mean} \in \mathbb{R}^3$

Figure 1. The structure of the proposed channel attention mechanism for the point cloud convolutional layer. The numbers in the parentheses show the size of each tensor before and after each operation. In this sample network, $|P| = 8$ and $n = 96$. The grey, red, blue, and green boxes represent max pooling, average pooling, the ConvPoint layer, and sigmoid operations, respectively. The C stands for concatenation.

In the next step, we propose to perform pooling on the points in the input points, i.e., $Pts = \left[\boldsymbol{p}_1\boldsymbol{p}_2\cdots\boldsymbol{p}_{|P|}\right]$, where $\boldsymbol{p}_i \in \mathbb{R}^3$ for $\boldsymbol{i} \in [\![1, |P|]\!]$ to create two points in the 3D space corresponding to the two features extracted from max and average pooling. The results for the two operations on Pts are $Pts_{max} \in \mathbb{R}^3$ and $Pts_{mean} \in \mathbb{R}^3$. The two vectors are concatenated using (5).

$$Pts^c = C(Pts_{max}, Pts_{mean}) \tag{5}$$

Pts^c has an appropriate size of $(2, 3)$ to be combined with X^c to create an input P^c to be applied to a ConvPoint layer, as shown in Figure 1. After passing the pairs of Pts^c and X^c to the Conv layer, the output will be passed to a σ function using (6).

$$\boldsymbol{M_c}(P) = \sigma(ConvPoint(Pts^c, X^c)) \tag{6}$$

The output of the attention block, i.e., $\boldsymbol{M_c}(P)$, is obtained by multiplying by the original input P using the element-wise multiplication in (3) and the results $\boldsymbol{F}'$ will be added to the original input.

3.4. The Network Structure

The network used in this project has a structure similar to U-Net, used in [9] for segmentation. The original network without the attention mechanism is given in Figure 2. The network is composed of two main parts, an encoder and a decoder. There are six ConvPoint layers in the encoder. Each ConvPoint layer is a part of six blocks, demonstrated in Figure 2, given by a number from (1) to (6). The output of each ConvPoint layer comprises point cloud data and it has two parts, i.e., the position of the points Pts_i and the corresponding features, i.e., X_i. The proposed attention module is applied to the network in different parts to find the best locations for the attention mechanism. For instance, it is applied after the final layer in the encoder just before entering the decoder, i.e., Pts_6 and X_6. The results were demonstrated in Section 4. The point cloud convolutional neural network with the proposed attention is called PointCloud-At.

Since input point clouds are given in different sizes, 2500 points are selected randomly, and the label of every point in the input is determined as output for segmentation. Cross-entropy loss is calculated for each point and the scores at the shape level are calculated accordingly. The scores for the network are the instance average intersection over union (mIoU) [9].

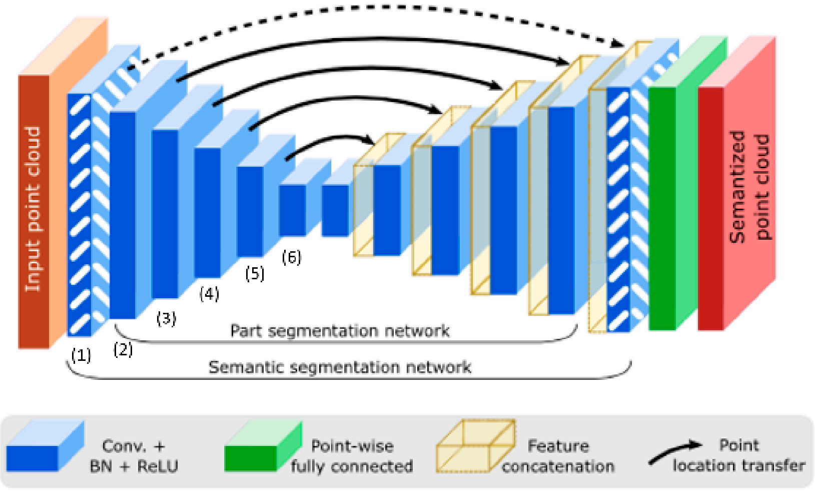


Figure 2. Semantic segmentation network graphical display, as fetched from [9].

4. Experiments and Results

Different experiments were carried out on the ShapeNet dataset [30] to thoroughly evaluate the proposed channel attention mechanism on point cloud data. These and the results are discussed in this section. The dataset is also described in Section 4.1.

4.1. Dataset

The proposed attention mechanism is evaluated on the ShapeNet dataset [30], which contains point clouds. Different experiments were run to evaluate the attention mechanisms from different perspectives of the ShapeNet dataset [30]. The ShapeNet dataset is a rich annotated 3D representation of shapes. It provides semantic annotations. The ShapeNet dataset contains 16,680 models that belong to 16 shape categories. These categories are split into training and testing sets, with each category annotated with two- to six-part labels. Thus, it has around fifty (50) classes [9].

4.2. Experiments

To evaluate the proposed attention module, we modified the attention module by positioning it in several areas in the network, including the skip connections. The results were compared with the results of the base method described in Section 4.3.

In the first experiment for the proposed method, the proposed attention mechanism is applied between the encoder and decoder of the network, which is a U-net. The proposed attention module shown in Figure 1 is applied on the final layer of the encoder, i.e., Pts_6 and X_6. The matrix shape of the output of each operation is shown in Figure 1. After completing these operations, the training was run for 10 epochs. There is an improvement in the mean intersection over union (mIoU) score compared to the score of the original network without the proposed attention module.

In the next step, we applied the attention module on (Pts_6, X_6) and (Pts_5, X_5), i.e., the outputs of blocks 6 and 5 in Figure 2. The improvement in this was higher compared to the previous experiment, hence making it a better technique. Results are shown in the following sections. The best model was found, and it was trained for 200 epochs to be compared with the base method that was trained for the same number of epochs.

4.3. The Results of the Base Method

Initially, the base network in [9] was evaluated before applying the proposed attention module. The base method, i.e., ConvPoint, has shown a competitive performance compared to state-of-the-art methods [9] when it trains for 200 epochs. The framework is ranked amongst the best five frameworks (ranked number four) for both mean class intersection over union and mean intersection over union (mcIoU and mIoU) on the ShapeNet dataset [9]. We initially trained the base method for ten epochs using the initial set, and the final mean intersection over union (mIoU) for all shapes was 80.3% (Table 1).

Table 1. The performance of the proposed method when the attention mechanism is added in different positions of the base network, which is compared with the base method, i.e., ConvPoint [9]. The training was performed for 10 epochs for all the methods.

Network/Attention	Number of Epochs	mIoU Score
Base method	10	80.3%
$(Pts_6\,,\ X_6)$	10	80.5%
$(Pts_5\,,\ X_5)$	10	80.4%
$(Pts_4\,,\ X_4)$	10	80.39%
(Pts_6, X_6) and (Pts_5, X_5)	10	81.45%
$(Ptse_5\,,\ Xe_5)$	10	80.30%

4.4. The Results of the Proposed Method

The proposed point cloud convolutional neural network with attention, called the PointCloud-At method, was trained for ten epochs like the base method for a fair comparison. A low number of training epochs was used to reduce computation time while testing different situations to find appropriate hyperparameters/structures.

In the first experiment, we applied attention to (Pts_6, X_6) and a score of 80.5% for the mIoU on all shapes was achieved. Secondly, we applied the attention mechanism to (Pts_5, X_5), which gave a mIoU of 80.4%. In the next experiment, the proposed attention module was applied to (Pts_4, X_4), and a mIoU score of 80.39% was achieved. Although different experiments showed improvement in the results compared to the base model, different combinations are tested to obtain a better score. Hence, the attention module is applied to the following outputs of the ConvPoint layers: (Pts_6, X_6) and (Pts_5, X_5). The mean intersection over union (mIoU) score of 81.45% was achieved, which is a better improvement compared to the previous cases. The results are given in Table 1.

In the next experiment, the attention mechanism was applied to the escape connection that connects the output of the fifth ConvPoint layer in the encoder to the corresponding layer in the decoder (Figure 2), i.e., ($Ptse_5$, Xe_5). A mean intersection over union score of 80.30% was obtained, which is close to the score of the base method, as shown in the last row of Table 1. The simulation results in Table 1 show that attention modules applied to (Pts_6, X_6) and(Pts_5, X_5) simultaneously could reach the best results. In the next experiments, the proposed method with the best results was trained for 200 epochs and compared to the base method trained for the same number of epochs (Table 2). Additionally, we explored using median pooling instead of max pooling, and the results are reported in the last row of Table 2.

Table 2. Comparison of the proposed method with the base method when they are trained for 200 epochs. The attention mechanism was applied to (Pts_6, X_6) and (Pts_5, X_5).

Network/Attention	Number of Epochs	mIoU Score
Base method	200	83%
Proposed method	200	84.2%
Proposed method using median pooling	200	84.2%

In our research, we initially used max pooling in our attention mechanism; this is a common practise in many deep learning and attention mechanism architectures. We thought that different pooling operations might capture different aspects of the point cloud; hence, we explored median pooling. Unlike max pooling, which selects the maximum value, median pooling measures and selects the middle value. Using both max and median pooling resulted in the same score, showing that they both can perform on the same level in this case, providing flexibility on which operation to use. Obtaining the same score further indicates the robustness of our attention mechanism and its stability.

4.5. Comparison with State-of-the-Art Frameworks

Having carefully studied the quantitative results of various frameworks, as reported in [31], ConvPoint [9] has shown a comparative performance compared to the state-of-the-art methods. In this paper, an attention mechanism was proposed to improve the performance of ConvPoint [9]. The comparison of the proposed method with the other state-of-the-art methods is shown in Table 3. The results in Table 3 show that the proposed method in this research equates to several frameworks in some cases or outperforms them in other cases.

In this research, an attention mechanism was proposed to be added to existing CNNs for point clouds. The proposed attention mechanism was integrated into a recent method called ConvPoint [9], enhancing the method's ability to focus on points containing important information. Consequently, the proposed method increased the mIoU score of ConvPoint [9] from 83.2 to 84.2.

Seventeen SotA methods are compared with the proposed method in Table 3. The proposed method has an accuracy higher than 15 SotA methods. While the base method, i.e., ConvPoint [9], has a mIoU score lower than SubSparseCN [32] and SPLATNet [33], applying the proposed attention method increased the accuracy of the proposed method to a value higher than the accuracies of SubSparseCN [32] and SPLATNet [33] that shows the effectiveness of the proposed method. The result in Table 3 shows the importance of the attention mechanism in deep neural networks and that applying the attention mechanism method to the other SotA methods can improve their accuracy.

Table 3. Comparison of the proposed method, i.e., PointCloud-At, with state-of-the-art methods.

Network	mIoU Score
PointCloud-At	84.2
SyncSpecCNN [34]	82.0
Pd-Network [35]	82.7
3DmFV-Net [36]	81.0
PointNet [17]	80.4
PointNet++ [14]	81.9
SubSparseCN [32]	83.3
SPLATNet [33]	83.7
SpiderCNN [37]	81.7
SO-Net [25]	81.0
PCNN [38]	81.8
KCNet [26]	82.2
RSNet [39]	81.4
DGCNN [15]	82.3
SGPN [40]	82.8
PointCNN [41]	84.6
KPConv [42]	85.1
ConvPoint [9]	83.2

Although KPConv [42] achieves the highest accuracy, it has a higher computational cost compared to the proposed method. The model size of KPConv was reported in [42]. The report shows that KPConv has 14.2 M parameters. However, the number of parameters in the proposed model is 1.3 M, which is much lower than the number of parameters in KPConv. The result shows that the number of parameters in the proposed model is about 11 times less than that of KPConv. The lower number of parameters in the proposed method makes it suitable for edge devices with limited computational resources and memory. Additionally, the low number of parameters reduces energy consumption. The results are shown in Table 4.

Table 4. Comparison of the number of parameters of the proposed method with KPConv [42], the method that achieves the highest accuracy.

Network	Number of Parameters
PointCloud-At	1.3 M
KPConv [42]	14.2 M

The proposed method uses a simple and efficient convolutional layer tailored for point clouds, and it is lightweight. It uses one kernel weight for each kernel point. However, KPConv [42] has a complex kernel filter, and each kernel point has a set of weights. Consequently, KPConv has a higher number of parameters compared to the proposed method.

5. A Discussion of the Applications of Point Cloud Processing

There is a growing demand for point cloud data processing in various applications that depend on 3D sensor data. Point cloud processing is a powerful tool with a wide range of applications across different scenarios in autonomous driving, robotics, virtual and augmented reality, medical imaging, digital surface modelling, automated building extraction, urban planning and visualisation, geographic information systems (GISs), and 3D modelling. The classification of dense point cloud data has been critical in creating detailed 3D models that have improved urban planning and development, medical imaging, autonomous driving, and many other fields.

Kurdi et al. [43] have used light detection and ranging (LiDAR) sensors for remote sensing applications. They demonstrated the potential of point cloud processing in urban planning. The authors proposed a method for automatic building point cloud filtering. The method divides building point clouds into different zones and extracts high tree crowns obstructing building structures. This application highlights the importance of processing point clouds extracted from 3D sensors for solving complex issues faced in urban planning, environmental management, and disaster management.

Maltezos et al. [44] explored point cloud processing in identifying several urban features. This work focused on improving the performance of building classification and extraction from densely populated areas, highlighting the technology's ability to handle complex urban environments. Furthermore, ref. [45] discussed how the automation of extracting buildings from LiDAR data streamlines the creation of digital surface models (DSMs). Their work emphasises that such automation is vital for a range of applications, from smart city development to cartographic analysis, and it shows the wide impact of point cloud processing in urban planning and geospatial intelligence.

These applications collectively highlight the importance of point cloud processing in modern life. For instance, by providing detailed and accurate 3D representations of complex environments, this technology supports more informed decision-making in infrastructure management and urban planning, paving the way for smarter and more efficient cities in the future.

6. Conclusions, Limitations, and Future Work

This research proposed a deep point convolutional neural network for point cloud data using an attention mechanism. The study used an attention mechanism designed for point clouds to improve the performance of the network. The attention mechanism works using a channel attention module. The channel attention module was proposed specifically for point cloud data with inspiration from the CBAM [21] which is for ordinary convolutional layers acting on regular matrices (2D images). The proposed method overcomes the difficulties in processing scattered point cloud data compared to usual image or voxel data, which have regular shapes. In the proposed attention method, average pooling and max pooling are performed on the points in 3D space to focus on the informative parts of the data. Through several experiments and evaluations, we have shown that our proposed method enhances the performance of the base framework.

In this research, we designed a channel attention mechanism to boost the performance of ConvPoint [9]. The proposed method uses a max pooling operation on both the features and the positions of its input points in the 3D space. Additionally, it uses average pooling on the features and the position of its input points. The two operations create two points in the 3D space with their corresponding features. Then, the two points are applied to a ConvPoint [9] layer to create the outputs of the attention block. The ConvPoint [9] layer is a convolutional layer that is designed to operate on point clouds. Therefore, the proposed attention mechanism not only has a specific pooling operation on the input points (features and location of the inputs) but also contains a ConvPoint [9] layer that extracts appropriate outputs for the attention mechanism to be multiplied by the original input. Note that the proposed attention mechanism has learning parameters in the ConvPoint [9] layer, and they are adjusted during training to create a reliable attention mechanism using the training data. The proposed attention module is applied to the U-Net in different parts. These unique properties make this method different from other attention mechanisms, such as the method proposed in [28].

In [28], a convolutional layer called LAE-Conv layer was proposed to apply to point clouds. Whereas the ConvPoint [9] layer was used in our approach as a base method, in [28], a pointwise spatial attention module was proposed to capture the global dependencies. They used MLP layers in the attention block. Only features of the points are applied to the MLP and the positions of the points were not used. However, in our method, a new channel attention mechanism was proposed to put appropriate weights on the channel of the input point clouds. Additionally, it uses a ConvPoint [9] layer (instead of an MLP) inside the attention block that considers the position of the input points in addition to their features. The proposed attention block is designed in such a way that can be used with different convolutional layers to determine an output that corresponds to the importance of each channel.

In this project, ConvPoint [9] was used as the base method. ConvPoint [9] is an end-to-end deep neural network for classification and segmentation. To the best of our knowledge, there was not a report of sensible failure in the base network. Our proposed attention mechanism was added to this base model, and we did not see any sensible failure case. Understanding the failure case is important, especially in generative models, such as generative adversarial networks (GANs), where two networks compete [46]. If the proposed method is used as a generator of a GAN in future work, then the failure cases need to be analysed.

Adding the proposed attention method to a base method improved its performance compared to the base method. The proposed method was compared with other state-of-the-art (SotA) methods. While the proposed framework does not significantly outperform all existing SotA methodologies, it does achieve a competitive result, matching and surpassing many established frameworks; this shows that the proposed method is valid and contributes meaningfully to the field. It shows clear improvement over the base method, indicating that the attention mechanism does enhance performance. It is designed to be easily integrated into existing point cloud CNNs, allows easy adoption in various architectures, and works directly on the point cloud.

The proposed attention mechanism uses max and average pooling operations, and the pooling operations enhance feature aggregation. The max pooling captures the most prominent features, which helps the network focus on the critical areas of the data. The average pooling reduces noise and improves generalisation, making the proposed approach better than existing ones. With the proposed method, the key aspects of the data are captured.

Whilst the proposed attention model used average pooling and max pooling along the channel axis to extract what the informative and vital inputs are, it did not explore the spatial axis to extract where the key elements are. Hence, it guides the framework to which channel to look in; however, it does not direct the network to where the vital elements are in space. Adopting a spatial attention module to enhance the framework is worth considering in future work. This will help the framework to focus on where in the space is important in the input point cloud features. Additionally, as the proposed attention method is designed to directly work with point cloud data, it can be applied to other different deep neural networks that are working directly with point cloud data.

Author Contributions: Conceptualization, S.U. and A.T.; methodology, S.U. and A.T.; software, S.U. and A.T.; validation, S.U. and A.T.; formal analysis, S.U. and A.T.; investigation, S.U. and A.T.; resources, A.T.; data curation, S.U. and A.T.; writing—original draft preparation, S.U.; writing—review and editing, A.T.; visualization, S.U. and A.T.; supervision, A.T.; project administration, A.T.; funding acquisition, A.T. All authors have read and agreed to the published version of the manuscript.

Funding: This study and work were supported by the Institute of Artificial Intelligence at De Montfort University Leicester, UK, through HIEF Enterprise Funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The results of original contributions presented in the study are included in the article, further inquiries can be directed to the authors.

Acknowledgments: We also appreciate John Files and James Brown from CEM for supporting us with HPC for processing deep neural networks.

Conflicts of Interest: The authors have no conflicts of interest relevant to this article.

References

1. Pepe, M.; Alfio, V.S.; Costantino, D. Rapid and Accurate Production of 3D Point Cloud via Latest-Generation Sensors in the Field of Cultural Heritage: A Comparison between SLAM and Spherical Videogrammetry. *Heritage* **2022**, *5*, 1910–1928. [CrossRef]
2. Bello, S.A.; Yu, S.; Wang, C.; Adam, J.M.; Li, J. Review: Deep learning on 3D point clouds. *Remote Sens.* **2020**, *12*, 1729. [CrossRef]
3. Taherkhani, A.; Cosma, G.; McGinnity, T.M. Deep-FS: A feature selection algorithm for Deep Boltzmann Machines. *Neurocomputing* **2018**, *322*, 22–37. [CrossRef]
4. Taherkhani, A.; Cosma, G.; McGinnity, T.M. AdaBoost-CNN: An adaptive boosting algorithm for convolutional neural networks to classify multi-class imbalanced datasets using transfer learning. *Neurocomputing* **2020**, *404*, 351–366. [CrossRef]
5. Alani, A.A.; Cosma, G.; Taherkhani, A.; McGinnity, T.M. Hand Gesture Recognition Using an Adapted Convolutional Neural Network with Data Augmentation. In Proceedings of the 2018 4th International Conference on Information Management (ICIM 2018), Oxford, UK, 25–27 May 2018; pp. 5–12. [CrossRef]
6. Taherkhani, A.; Cosma, G.; Alani, A.A.; McGinnity, T.M. Activity recognition from multi-modal sensor data using a deep convolutional neural network. *Adv. Intell. Syst. Comput.* **2019**, *857*, 203–218. [CrossRef]
7. Guo, Y.; Wang, H.; Hu, Q.; Liu, H.; Liu, L.; Bennamoun, M. Deep Learning for 3D Point Clouds: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 4338–4364. [CrossRef]
8. Minaee, S.; Boykov, Y.; Porikli, F.; Plaza, A.; Kehtarnavaz, N.; Terzopoulos, D. Image Segmentation Using Deep Learning: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *44*, 3523–3542. [CrossRef]
9. Boulch, A. ConvPoint: Continuous convolutions for point cloud processing. *Comput. Graph.* **2020**, *88*, 24–34. [CrossRef]
10. Niu, Z.; Zhong, G.; Yu, H. A review on the attention mechanism of deep learning. *Neurocomputing* **2021**, *452*, 48–62. [CrossRef]
11. Liu, T.; Luo, R.; Xu, L.; Feng, D.; Cao, L.; Liu, S.; Guo, J. Spatial Channel Attention for Deep Convolutional Neural Networks. *Mathematics* **2022**, *10*, 1750. [CrossRef]
12. Zhang, Q.L.; Yang, Y.B. SA-Net: Shuffle Attention for Deep Convolutional Neural Networks (ICASSP). In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing-Proceedings, Toronto, ON, Canada, 6–11 June 2021; Volume 2021, pp. 2235–2239. [CrossRef]
13. Li, Y.; Ma, L.; Zhong, Z.; Liu, F.; Chapman, M.A.; Cao, D.; Li, J. Deep Learning for LiDAR Point Clouds in Autonomous Driving: A Review. *IEEE Trans. Neural. Netw. Learn Syst.* **2021**, *32*, 3412–3432. [CrossRef] [PubMed]
14. Qi, C.; Yi, L.; Su, H.; Guibas, L. PointNet++: Deep Hierarchical Feature Learning on. In Proceedings of the NIPS'17: 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4 February 2017; pp. 5105–5114.
15. Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S.E.; Bronstein, M.M.; Solomon, J.M. Dynamic Graph CNN for Learning on Point Clouds. *ACM Trans. Graph.* **2018**, *38*, 13. [CrossRef]
16. Zhao, H.; Jiang, L.; Fu, C.-W.; Jia, J. PointWeb: Enhancing Local Neighborhood Features for Point Cloud Processing. Available online: https://github.com/hszhao/PointWeb (accessed on 7 November 2022).
17. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, 21–26 July 2017; Volume 2017, pp. 77–85. [CrossRef]

18. Hua, B.-S.; Tran, M.-K.; Yeung, S.-K. Pointwise Convolutional Neural Networks. Available online: http://arxiv.org/abs/1712.05245 (accessed on 14 December 2017).
19. Lei, H.; Akhtar, N.; Mian, A. Octree Guided CNN with Spherical Kernels for 3D Point Clouds. Available online: http://arxiv.org/abs/1903.00343 (accessed on 28 February 2019).
20. Liu, Y.; Fan, B.; Xiang, S.; Pan, C. Relation-Shape Convolutional Neural Network for Point Cloud Analysis. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.
21. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. *CBAM: Convolutional Block Attention Module*; Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); Springer: Berlin/Heidelberg, Germany, 2018; Volume 11211, pp. 3–19. [CrossRef]
22. Yang, Y.; Ma, Y.; Zhang, J.; Gao, X.; Xu, M. Attpnet: Attention-based deep neural network for 3D point set analysis. *Sensors* **2020**, *20*, 5455. [CrossRef] [PubMed]
23. Hu, Z.; Zhang, D.; Li, S.; Qin, H. Attention-based relation and context modeling for point cloud semantic segmentation. *Comput. Graph.* **2020**, *90*, 126–134. [CrossRef]
24. Deng, S.; Dong, Q. GA-NET: Global Attention Network for Point Cloud Semantic Segmentation. *IEEE Signal Process. Lett.* **2021**, *28*, 1300–1304. [CrossRef]
25. Li, J.; Chen, B.M.; Lee, G.H. SO-Net: Self-Organizing Network for Point Cloud Analysis. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 9397–9406. [CrossRef]
26. Shen, Y.; Feng, C.; Yang, Y.; Tian, D. Mining Point Cloud Local Structures by Kernel Correlation and Graph Pooling. 2018. Available online: http://www.merl.com/research/ (accessed on 7 November 2022).
27. Cui, Y.; An, Y.; Sun, W.; Hu, H.; Song, X. Lightweight Attention Module for Deep Learning on Classification and Segmentation of 3-D Point Clouds. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–12. [CrossRef]
28. Feng, M.; Zhang, L.; Lin, X.; Gilani, S.Z.; Mian, A. Point attention network for semantic segmentation of 3D point clouds. *Pattern Recognit.* **2020**, *107*, 107446. [CrossRef]
29. Landrieu, L.; Simonovsky, M. Large-Scale Point Cloud Semantic Segmentation with Superpoint Graphs. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4558–4567. [CrossRef]
30. Chang, A.X.; Funkhouser, T.; Guibas, L.; Hanrahan, P.; Huang, Q.; Li, Z.; Savarese, S.; Savva, M.; Song, S.; Su, H.; et al. ShapeNet: An Information-Rich 3D Model Repository. *arXiv* **2015**. [CrossRef]
31. Hu, Q.; Yang, B.; Xie, L.; Rosa, S.; Guo, Y.; Wang, Z.; Trigoni, N.; Markham, A. Learning Semantic Segmentation of Large-Scale Point Clouds with Random Sampling. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 1–20. [CrossRef]
32. Graham, B.; Engelcke, M.; Van Der Maaten, L. 3D Semantic Segmentation with Submanifold Sparse Convolutional Networks. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 9224–9232. [CrossRef]
33. Su, H.; Jampani, V.; Sun, D.; Maji, S.; Kalogerakis, E.; Yang, M.H.; Kautz, J. SPLATNet: Sparse Lattice Networks for Point Cloud Processing. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2530–2539. [CrossRef]
34. Yi, L.; Su, H.; Guo, X.; Guibas, L.J. SyncSpecCNN: Synchronized Spectral CNN for 3D Shape Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
35. Klokov, R.; Lempitsky, V. Escape from Cells: Deep Kd-Networks for the Recognition of 3D Point Cloud Models. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; Volume 2017, pp. 863–872. [CrossRef]
36. Ben-Shabat, Y.; Lindenbaum, M.; Fischer, A. 3D Point Cloud Classification and Segmentation using 3D Modified Fisher Vector Representation for Convolutional Neural Networks. *arXiv* **2017**. [CrossRef]
37. Xu, Y.; Fan, T.; Xu, M.; Zeng, L.; Qiao, Y. SpiderCNN: Deep Learning on Point Sets with Parameterized Convolutional Filters. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; Volume 11212, pp. 90–105. [CrossRef]
38. Atzmon, M.; Maron, H.; Lipman, Y. Point Convolutional Neural Networks by Extension Operators. *arXiv* **2018**. [CrossRef]
39. Huang, Q.; Wang, W.; Neumann, U. Recurrent Slice Networks for 3D Segmentation of Point Clouds. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2626–2635. [CrossRef]
40. Wang, C.; Samari, B.; Siddiqi, K. Local Spectral Graph Convolution for Point Set Feature Learning. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; Volume 11208, pp. 56–71. [CrossRef]
41. Li, Y.; Bu, R.; Sun, M.; Wu, W.; Di, X.; Chen, B. PointCNN: Convolution On X-Transformed Points. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 3–8 December 2018.
42. Thomas, H.; Qi, C.R.; Deschaud, J.E.; Marcotegui, B.; Goulette, F.; Guibas, L. KPConv: Flexible and Deformable Convolution for Point Clouds. In Proceedings of the IEEE International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; Volume 2019, pp. 6410–6419. [CrossRef]

43. Tarsha Kurdi, F.; Gharineiat, Z.; Campbell, G.; Awrangjeb, M.; Dey, E.K. Automatic Filtering of Lidar Building Point Cloud in Case of Trees Associated to Building Roof. *Remote Sens.* **2022**, *14*, 430. [CrossRef]
44. Maltezos, E.; Doulamis, A.; Doulamis, N.; Ioannidis, C. Building extraction from LiDAR data applying deep convolutional neural networks. *IEEE Geosci. Remote Sens. Lett.* **2019**, *16*, 155–159. [CrossRef]
45. Bethel, J.; Elaksher, A.F.; Bethel, J.S. Reconstructing 3D Buildings from LiDAR Data. Available online: https://www.researchgate.net/publication/228777898 (accessed on 7 November 2022).
46. Li, W.; Fan, L.; Wang, Z.; Ma, C.; Cui, X. Tackling mode collapse in multi-generator GANs with orthogonal vectors. *Pattern Recognit.* **2021**, *110*, 107646. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

sensors

MDPI

Article

Automated Phenotypic Trait Extraction for Rice Plant Using Terrestrial Laser Scanning Data

Kexiao Wang *, Xiaojun Pu and Bo Li

Institute of Agricultural Science and Technology Information, Chongqing Academy of Agricultural Sciences, Chongqing 401329, China; puxiaojunanna@163.com (X.P.); hnjzbobo@163.com (B.L.)
* Correspondence: wangkexiao_2007@126.com

Abstract: To quickly obtain rice plant phenotypic traits, this study put forward the computational process of six rice phenotype features (e.g., crown diameter, perimeter of stem, plant height, surface area, volume, and projected leaf area) using terrestrial laser scanning (TLS) data, and proposed the extraction method for the tiller number of rice plants. Specifically, for the first time, we designed and developed an automated phenotype extraction tool for rice plants with a three-layer architecture based on the PyQt5 framework and Open3D library. The results show that the linear coefficients of determination (R^2) between the measured values and the extracted values marked a better reliability among the selected four verification features. The root mean square error (RMSE) of crown diameter, perimeter of stem, and plant height is stable at the centimeter level, and that of the tiller number is as low as 1.63. The relative root mean squared error (RRMSE) of crown diameter, plant height, and tiller number stays within 10%, and that of perimeter of stem is 18.29%. In addition, the user-friendly automatic extraction tool can efficiently extract the phenotypic features of rice plant, and provide a convenient tool for quickly gaining phenotypic trait features of rice plant point clouds. However, the comparison and verification of phenotype feature extraction results supported by more rice plant sample data, as well as the improvement of accuracy algorithms, remain as the focus of our future research. The study can offer a reference for crop phenotype extraction using 3D point clouds.

Keywords: phenotypic parameters; rice plant; automatic extraction; 3D point cloud; terrestrial laser scanning (TLS)

1. Introduction

Plant phenotypes, with genetic and environmental factors, are commonly used by plant breeders to meet specific breeding goals [1]. Quantitative evaluation of crop phenotype characteristics is an important aspect of crop breeding research, and is crucial for revealing the mechanism of biological traits [2,3]. The bottleneck of plant phenotype research lies in determining how to quickly obtain enough plant phenotype features [4]. Traditional methods for extracting crop plant phenotypic features often need manual measurement, which are time-consuming and labor intensive [5]. In recent years, traditional radiation transmittance measurements have been widely applied in related research, but the sensors could only obtain two-dimensional (2D) images of the target crop, thus gaining limited phenotypic information [6].

The emergence of light detection and ranging (LiDAR) technology, which can generate the accurate three-dimensional (3D) information of the target, has provided technical support for obtaining the structural information of scanning targets at a higher level of detail [7]. Currently, its scope has spread across a wide range of research areas, especially in agriculture [8]. As one of the agricultural application fields of LiDAR, crop phenotype feature extraction had also become a research focus of many scholars. Among the research on crop population phenotypes, many studies concentrated on crop canopy height [9–11], canopy biomass [12–14], and leaf area index (LAI) [15,16] using unmanned aerial vehicle

(UAV) systems. Luo et al. estimated maize LAI, canopy height, and aboveground biomass using the combined hyperspectral imagery and LiDAR pseudo-waveforms, showing the strong liner correlation between LiDAR variables and LAI, height, and biomass [17]. Zhou et al. obtained the height of the maize canopy using a canopy height model based on the UAV-LiDAR data at different phases with higher estimation accuracy [18]. Compared with airborne LiDAR, terrestrial laser scanning (TLS) could provide precise, high-density, and repeatable data acquisition for monitoring specific crops with a cost-effective and easy to perform approach [19], which has been widely used in the precise extraction of individual crop phenotypic features. Shi et al. automatically measured the corn plant location and spacing by TLS, with a total plant counting error of 5.5% and a RMSE in spacing measurement of 1.9 cm [20]. Guo et al. estimated the wheat height using TLS, pointing out the 95th height percentile, H_{95}, can effectively monitor the height during the entire growth stages, and at which the wheat height can be accurately detected for heights as low as 0.18 m [21]. Jin et al. proposed a median normalized-vector growth (MNVG) algorithm, which can segment the stem and leaf of the maize samples with different heights, degrees of compactness, leaf numbers, and densities from three growing stages using terrestrial LiDAR data [22]. Su et al. calculated plant height, plant area index (PAI), and projected leaf area (PLA) from the point clouds of different maize varieties under drought stress collected by TLS at the individual plant level [23]. There are many related studies of this kind, which have shown great potentiality in high-precision acquisition of crop features. However, relevant research has mostly focused on the population characteristics at the canopy level of crops, as well as the extraction of individual phenotypic features of plants such as maize, rapeseed [24], wheat [25], cabbage [26], and soybean [27]. As one of the important food crops, rice plays an important role in preserving food security and social stability. The rice plant has flat leaves, numerous tiller branches, and complex canopy structures, which pose significant difficulties in extracting its phenotypic features. Therefore, accurate extraction of rice plant phenotypes is of great significance in monitoring rice growth and crop breeding.

In this study, we put forward the automatic computational process of seven rice phenotype features by introducing multi-step point cloud processing algorithms, developed a plant phenotype feature extraction tool based on high-precision TLS data, and achieved the automatic extraction of rice plant parameters. This study can provide a reference for the analysis of rice crop characteristics. The main contributions of this study are as follows:

(1) Giving an automatic calculation acquisition approach to obtain the phenotype features of rice plants from a rice plant point cloud, including crown diameter, perimeter of stem, plant height, surface area, plant volume and PLA, and tiller number. We also proposed a point cloud extraction method for the tiller number for rice plants.

(2) Combined with PyQt5 and the Open3D library for processing 3D point cloud data, and integrating the related point cloud data algorithms, we purposefully designed the first tool for automatically obtaining phenotype parameters of rice plants, which is conducive to promoting the application of 3D point cloud technology in rice plant breeding.

2. Indicator Determination

Plant height, PLA, and plant volume are important metrics to estimate crop yield [26]. With the growth of the plant stem and increase in the tiller number, the plant canopy and leaf area accumulate a large amount of organic matter for the later nutritional and reproductive growth of rice. The plant surface area reflects the photosynthetic capacity of crop plants. Meanwhile, the crown diameter and the perimeter of the stem are closely related to plant biomass and plant growth status to some extent [28]. The tiller number of rice is an important factor reflecting the yield potential [29]. Therefore, combining the characteristics of the rice plant, this study selected seven phenotype features closely related to the rice plant biomass, photosynthesis, and yield as the phenotype target features of the rice plant to carry out the tool design.

The feature indicators of the rice plant are shown in Figure 1. The crown diameter is the maximum radius of the plant canopy from the horizontal centroid, and PLA is the percentage of the horizontal projected area in the minimum bounding rectangle of the canopy (Figure 1a). In Figure 1b, the perimeter of stem and tiller number are obtained by calculating the perimeter and number of branches at the section of the plant stem. The plant height is the vertical distance from the highest point of the plant to the reference point (Figure 1c). The surface area and volume are the surface area of the 3D model of the plant body (Figure 1d) and the volume enclosed by the minimum convex hull (Figure 1e), respectively. Table 1 presents the specific indicator algorithm outline used in this study.

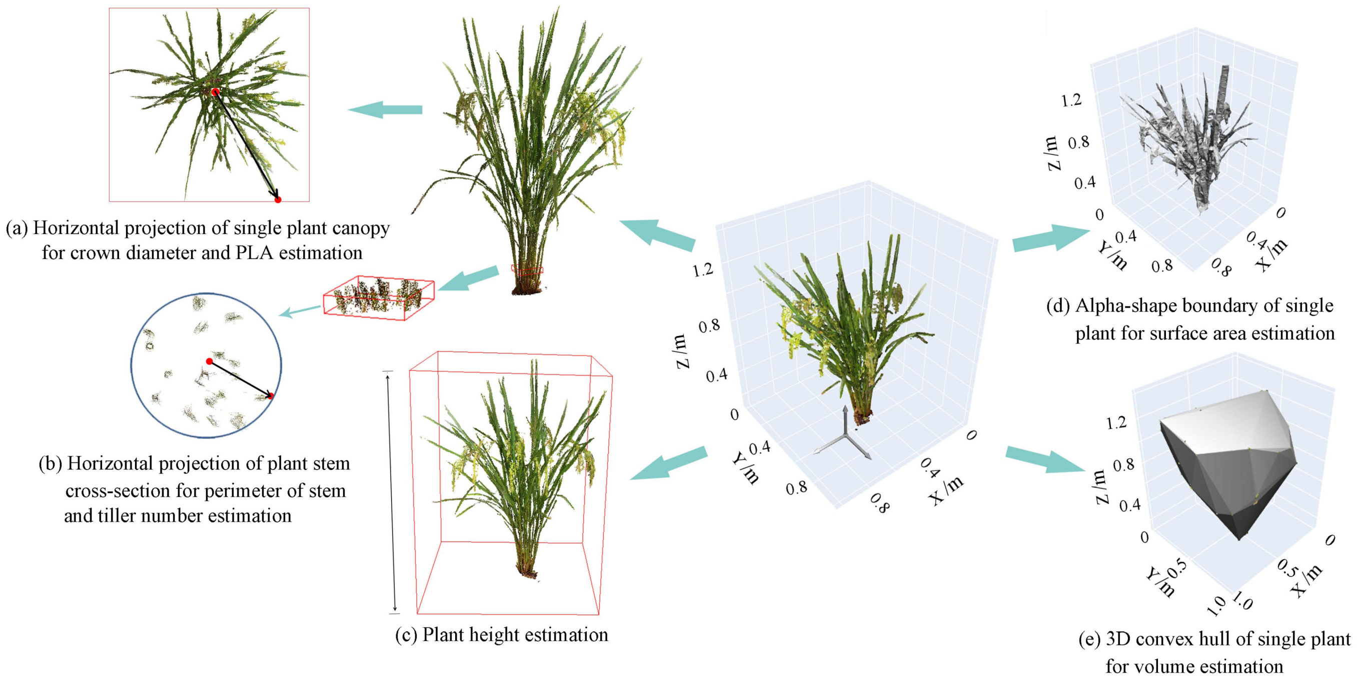

(a) Horizontal projection of single plant canopy for crown diameter and PLA estimation
(b) Horizontal projection of plant stem cross-section for perimeter of stem and tiller number estimation
(c) Plant height estimation
(d) Alpha-shape boundary of single plant for surface area estimation
(e) 3D convex hull of single plant for volume estimation
Z/m
Y/m
X/m
1.2
0.8
0.4
0
0.5
1.0

Figure 1. Schematic diagram of rice plant phenotypic feature.

Table 1. Algorithm approach of phenotypic indicators for rice plants.

Indicators	Algorithm Outline
Crown diameter	Search for the horizontal distance from horizontal centroid of the stem cross-section to the farthest point of the projection of the canopy.
Perimeter of stem	Horizontally cut the plant stem, select the target stem section and project it onto the horizontal plane. Then use "neighborhood search algorithm" to search for the horizontal distance from its centroid to the farthest point, and regard it as the radius to calculate the circumference.
Plant height	Obtain the vertical distance from the highest point of the plant to the reference point.
Surface area	Convert the point cloud of plant to a surface model, and then calculate its surface area by the Alpha-shape boundary algorithm.
Volume	Calculate the volume of the minimum convex hull that surrounds crop plant.
PLA	Calculate the projection area of plant using the grid method, and the minimum bounding rectangle area by oriented bounding box (OBB) of the canopy horizontal projection.
Tiller number	Cut the stem horizontally into slices to get point cloud data of the target stem section, carry statistical filtering and hierarchical density clustering segmentation and count.

3. Methodology

3.1. Overview

The flow chart of this study is presented in Figure 2. After the point cloud data collection and preprocessing, key algorithms of point cloud neighborhood search, Alpha-shape, and Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) were utilized to extract the relevant phenotype parameters. Meanwhile, the reliability validation was conveniently conducted on the extraction results of the four indicators for field measurement. Finally, a visual tool for automatic extraction of rice phenotypic parameters was designed based on the Open3D library and PyQt framework.

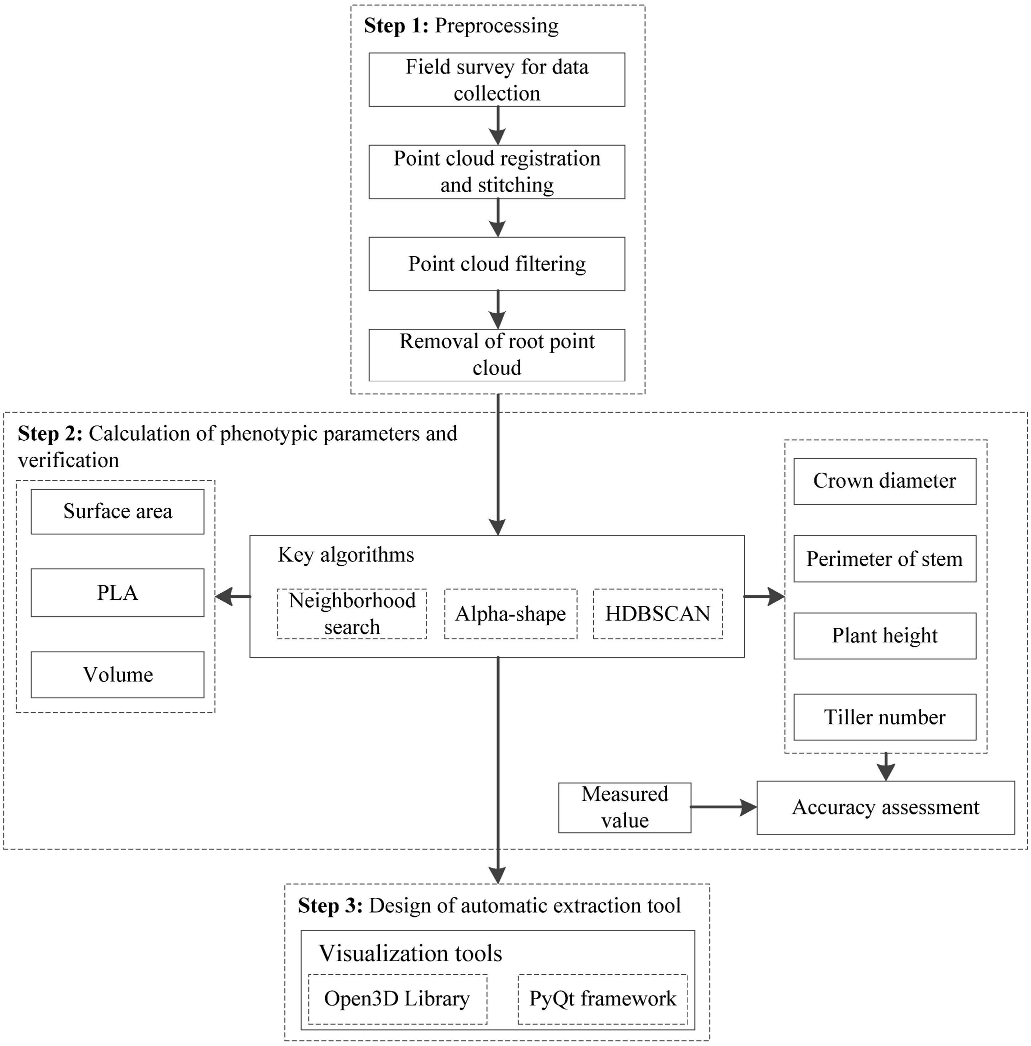


Figure 2. Flow chart of the study.

3.2. Data Acquisition and Processing

The point cloud data of rice plants were obtained indoors by the TLS equipment called FARO Focus S70, with an angle resolution of 0.009° and measurement error between 0.1 and 1.3 mm (Figure 3). To obtain the relatively complete plant point cloud data, the scanner was set up on both sides 1.5 m from the crop plants, with fixed scanning resolution of 1/4, quality level of 3× and scanning size of 10240 × 4267 Pt. Three target balls with a diameter of 15 cm were used for point cloud data registration. The registration process was completed in Faro Scene software 2020 with the fitting standard deviation of the balls within 1.5 mm. Then, objects unrelated to the plant were manually removed, and statistical outlier removal (SOR) filtering was used to clean the noise of the point cloud. Lastly, after removing the root point cloud, the “PCD cloud” format was exported for reading using Open3D library.

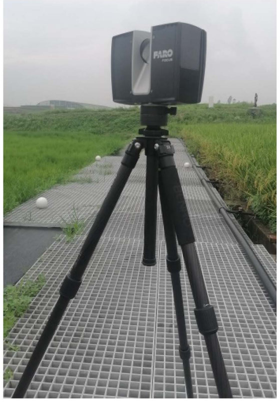


Figure 3. TLS sensor for data collection.

In addition, the crown diameter, perimeter of stem, and plant height of rice plants were obtained using a measuring tape, and the tiller number was obtained through manual counting. Assuming a measurement error of 1 cm, the average of three measured values was selected as the final value. The crown diameter was gained by measuring the diameter in both the maximum and minimum crown directions, calculating the average value, and converting it into a radius. The circumference of the stem was measured 3 cm above the root to obtain the perimeter of stem. Plant height was measured as the vertical distance from the root stem junction to the top of the plant.

3.3. Key Algorithms

3.3.1. Neighborhood Search

Kd-tree [30] is a binary search tree which is commonly used to establish high-dimensional spatial indexes in k-dimensional data space, to achieve the construction of geometric topology information between discrete point clouds and thereby find the nearest neighbors of query points. In this study, the neighborhood search algorithm was used to search for the distance to calculate the perimeter of the stem and crown diameter of rice plants. On the basis of constructing a point cloud Kd-tree, the geometric topological relationships between discrete points were established to conduct a point cloud neighborhood search. This study made use of the radius neighborhood search algorithm to traverse the centroid and all horizontal projection points with a step having a radius of 2 cm, and returned the search radius corresponding to its farthest point. The main flow of the radius neighborhood search module is shown in Algorithm 1.

Algorithm 1. Radius neighborhood search algorithm.

```
Input: The point cloud of stem cross-section or rice plant
1: Project the point data onto the X-Y plane
2: Calculate the centroid position of the horizontal projection point: $\mathbf{Z}(x_i, y_i)$
3: Count the horizontal projection point cloud: $n$
4: Build Kd-tree structure for point clouds
5: Set initial search radius and step size
6: for R in the radius range
7:     Neighborhood search with $Z(x_i, y_i)$ as the center and R as the radius, and record the number of points searched: k
8:     if $\mathbf{k} = n$:
9:         return R
10: end
Output: Calculate the perimeter of stem or crown diameter of rice plant
```

3.3.2. Three-Dimensional Surface Reconstruction

Alpha-shape [31] is a boundary point extraction algorithm proposed to construct a ball with the radius α based on three points in space. The ball is used to determine the boundary of the point cloud model. The basic Alpha-shape algorithm relies on the Delaunay triangulation, in which each triangle edge is characterized with a radius α of the smallest empty circle containing the edge or triangle [32]. Assuming that S is a finite point set in 3D space, and α is a real number ($\alpha \in [0, \infty)$). Alpha-shape is the convex hull of S point set when $\alpha = \infty$. Along with α value reduction, the ball gradually generates pits or holes, and the shape of the target tends to become more refined. This study mainly utilized the Alpha-shape algorithm to complete the 3D surface reconstruction process of rice plants and further extract the plant surface area. By comparison, the range of α was set from 0 to 2, with an initial value of 0.0040 and increasing step size of 0.0005. The 3D surface reconstruction effect of the rice plant corresponding to the value of 0.05, 0.01, and 0.005 is shown in Figure 4.

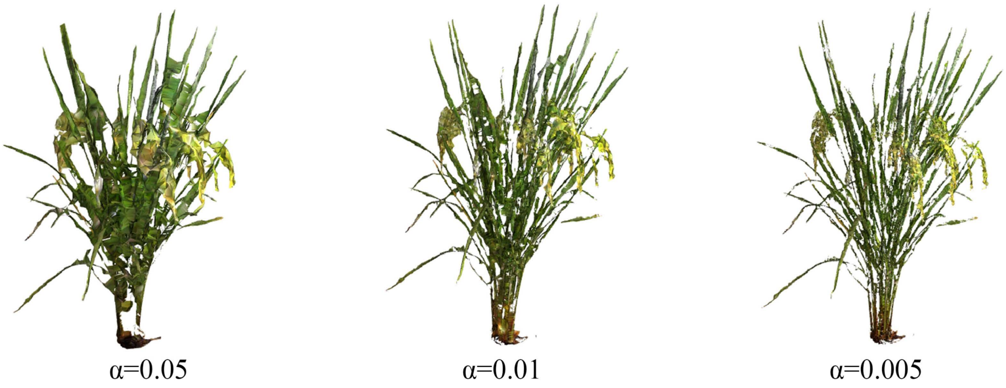


Figure 4. Three-dimensional reconstruction model of rice plants under different α values.

3.3.3. Point Cloud Density Clustering

The HDBSCAN algorithm is a data clustering method proposed by Campello [33], which combines both hierarchical clustering and density segmentation. It extends the Density-Based Spatial Clustering of Application with Noise (DBSCAN) algorithm by converting it into a hierarchical clustering algorithm to extract a flat clustering based on the stability of clusters, which is more robust to parameter selection. The basic principle is to construct a reachable graph by calculating the reachable distance between adjacent points and the core point, and finally introduce hierarchical clustering and cluster tree compression to obtain the final cluster. The reachable distance between two points is shown in Formula (1).

$$d_k(p,q) = max\{c_k(p), c_k(q), d(p,q)\} \tag{1}$$

where $d(p,q)$ is the original metric distance between p and q, and the core distance $c_k(p) = d\left(p, N^k(p)\right)$ represents the distance between the core point p and the k-th neighboring point.

Because of the tiller branches in the stem of the rice plant and the high-density point clouds, this study introduced the HDBSCAN algorithm to cluster the point clouds of the plant stem to obtain the tiller number. The algorithm was mainly implemented by calling the Python library “hdbscan” for point cloud data clustering programming. The implementation process can be decomposed into five steps [34]: (1) transform the space according to the density, (2) build the minimum spanning tree of the distance weighted graph, (3) construct a cluster hierarchy of connected components, (4) condense the cluster hierarchy based on the minimum cluster size, and (5) extract the stable clusters.

3.4. Accuracy Evaluation

In this study, a linear function was adopted to fit the relationships between the extracted values and manually measured values, and the statistical indicators R^2, *RMSE*, and *RRMSE* were utilized to assess the relationships. The model performance was more accurate with R^2 near to 1, and *RMSE* and *RRMSE* near to 0. *RRMSE* represents the degree of difference between the extracted and the measured values (*RRMSE* < 10% indicates no difference, 10% ≤ *RRMSE* < 20% denotes a small difference, 20% ≤ *RRMSE* < 30% is moderate, and *RRMSE* ≥ 30% represents a large difference [35]. The calculation formulas of R^2, *RMSE*, and *RRMSE* are shown in Formulas (2)–(4):

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(\hat{x}_i - \overline{x})^2}{\sum_{i=1}^{n}(x_i - \overline{x})^2} \tag{2}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\hat{x}_i - x_i)^2}{n}} \tag{3}$$

$$RRMSE = \frac{RMSE}{\overline{x}} \times 100\% \tag{4}$$

where x_i and $\overline{x}$ represent the measured value and the average of the measured values, respectively, $\hat{x}_i$ represents the extracted value, and n represents the number of samples.

3.5. Visualization Tools

3.5.1. Open3D Library

Open3D is a high-performance open-source library for fast processing of 3D point clouds developed by Intel Labs [36], supporting Python and C++ development environments. The main core functions of Open3D library include 3D data structures, 3D data processing algorithms, 3D scene reconstruction, 3D visualization, and 3D machine learning. The extraction of rice plant phenotype parameters in this visualization tool mainly involved its input/output module, geometric processing module, and visualization module.

3.5.2. PyQt Framework

Qt is a cross-platform C++ Graphical User Interface (GUI) application development framework, which integrates numerous form controls. PyQt is the Python interface of the graphical programming framework, which can call the Application Programming Interface (API) function for the GUI system design and development through Python. With the connection mechanism between signals and slots, PyQt5 integrates multiple classes, which are distributed across multiple functional modules. In this study, the design of GUI was completed by the Qt Designer tool, which mainly involved the object classes such as "QMainWindow" class, "QAlication class", "QDialog" class, and "QWidget" class. The modules mainly included "QtCore", "QtGui", "QtWidgets", "QFileDialog", and "QInputDialog". The form controls mainly included the "QMenuBar" control for displaying functional menus, the "QLabel" control for indicator labels, and the "QTextBrowser" control for extraction results.

4. Results

4.1. Extraction Result Validation

Due to the need for professional instruments to measure the indicators such as plant surface area, PLA, and plant volume, this study conveniently selected the four indicators for field measurement to verify the extraction accuracy. The measured values were employed as standard values to compare with the extracted results of the rice plant phenotype parameters. As shown in Figure 5, the R^2 values of the linear correlation between the measured value and the extracted values of crown diameter and tiller number are all above 0.80, and that of plant height is 0.97, which indicates the stronger correlation. The R^2

of perimeter of stem is 0.66, slightly lower than other indicators, which may be caused by the difference between selecting stem slices and manually measuring stem position. Table 2 gives the *RMSE* and *RRMSE* of the four indicators. The *RMSE* of crown diameter, perimeter of stem, and plant height are maintained at the level of centimeter, and the tiller number is only 1.63. The *RRMSE* of crown diameter, plant height, and tiller number stay within 10%, indicating no obvious difference. However, the value of perimeter of stem is 18.29%, a small difference from the actual observations, owing to the possible differences in measurement positions. On the whole, the extraction algorithm embedded in the tool exhibits better robustness and can accurately extract the corresponding phenotypic characteristic parameters of rice plants.

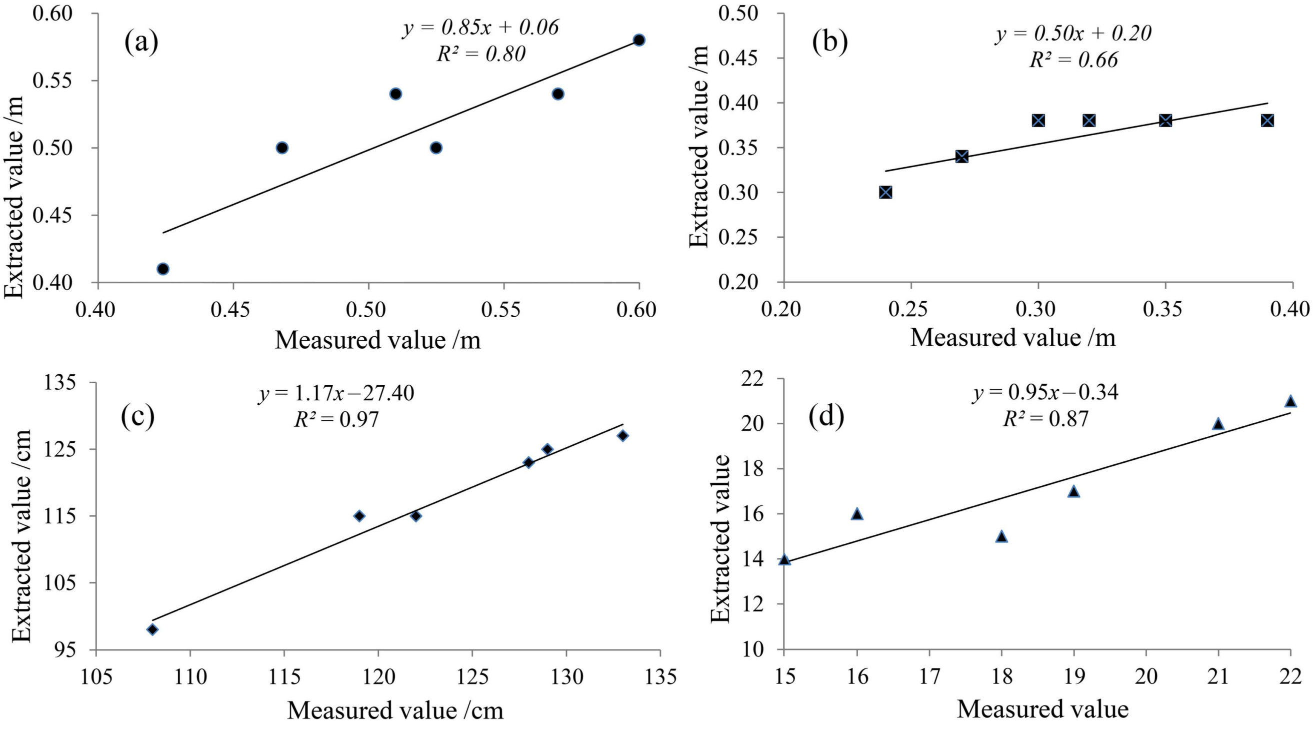


Figure 5. Scatter plots of the extracted and manually measured values of the rice plant. (**a**) Crown diameter; (**b**) perimeter of stem; (**c**) plant height and (**d**) tiller number.

Table 2. *RMSE* and *RRMSE* of various indicators in the study.

Indicators	*RMSE*	*RRMSE*
Crown diameter	0.03 m	5.03%
Perimeter of stem	0.06 m	18.29%
Plant height	6.35 cm	5.16%
Tiller number	1.63	8.82%

4.2. Implementation of Automatic Extraction Tool

4.2.1. Overall Structural Design

For automatic extraction of rice plant features and display, this study employed the technologies of fast processing of 3D point cloud data and computer visualization programming to build the plant parameter extraction tool. Following the principles of a simple interface and convenient operation, the architecture of the rice plant phenotype feature automatic extraction tool was divided into three parts: data interaction layer, basic processing layer, and functional module layer. The overall architecture is shown in Figure 6.

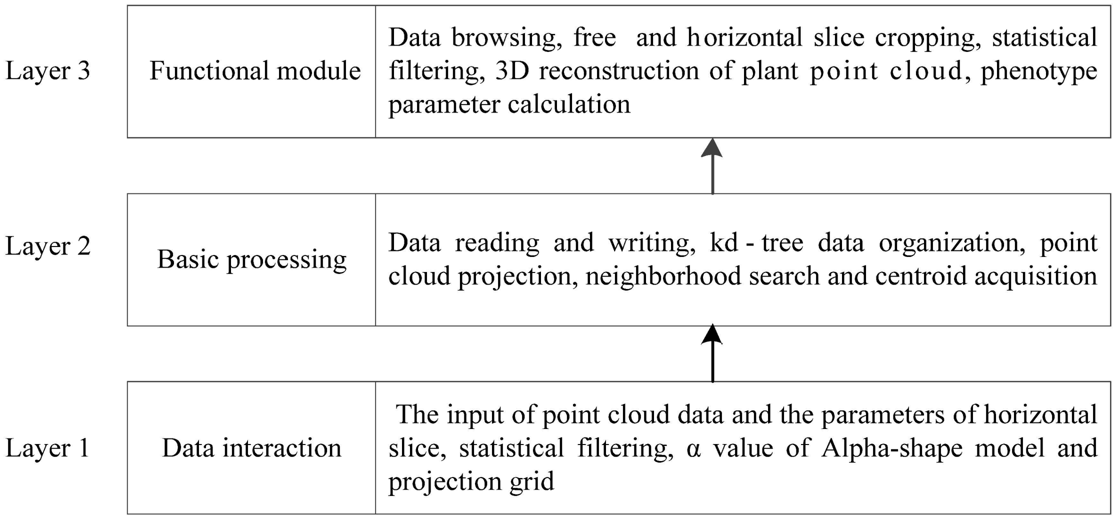

Layer 3
Functional module
Data browsing, free and horizontal slice cropping, statistical filtering, 3D reconstruction of plant point cloud, phenotype parameter calculation
Layer 2
Basic processing
Data reading and writing, kd - tree data organization, point cloud projection, neighborhood search and centroid acquisition
Layer 1
Data interaction
The input of point cloud data and the parameters of horizontal slice, statistical filtering, α value of Alpha-shape model and projection grid

Figure 6. Architecture of rice plant phenotypic parameter extraction system.

- The data interaction layer mainly enables necessary data entry. The input data are mainly the plant point cloud datasets and some function parameters, such as the slice height of the point cloud, α value of the Alpha-shape algorithm, and the grid size of point cloud projection.
- The basic processing layer mainly involves the processing of input point cloud data, including reading and writing, organization, plane projection transformation, and centroid acquisition of point cloud data. These processes mainly provide basic data for implementing the subsequent main functional modules.
- The functional module layer is the key of the automatic extraction of rice plant phenotype parameters. The functions include data visualization browsing, free cropping and horizontal slice cropping, statistical filtering, surface model reconstruction, and automatic calculation and display of plant phenotype parameters.

4.2.2. Division of Functional Module

Based on the architecture, the rice plant phenotype parameter extraction system can be divided into five menu functional modules: point cloud data browsing, data cropping, point cloud filtering, plant 3D reconstruction, and phenotype parameter calculation. The detailed description of each module is shown in Figure 7. After executing the corresponding module according to the requirements, automatic extraction and display of rice plant phenotype parameters can be achieved by inputting a small number of parameters.

4.2.3. Implementation of Automatic Extraction Function

The processing and visualization of point cloud data mainly relies on the Open3D library, machine learning scikit-learn library, and visualization graphics framework PyQt5. The development environment was the integrated development environment PyCharm 2023.2.1 under Win 10, paired with the open-source Python package manager "Anconda3".

- The parameter calculation module was mainly used for the calculation and display of phenotypic parameter results. The slot function included the parameter calculation process, and the corresponding label objects and display text box objects were designed to label and display the calculation results. During the execution process of indicator calculation, the corresponding phenotype parameters were automatically calculated and displayed in view of the input of plant point cloud data, stem trimming point cloud data, α value, and point cloud projection grid spacing (Figure 8).

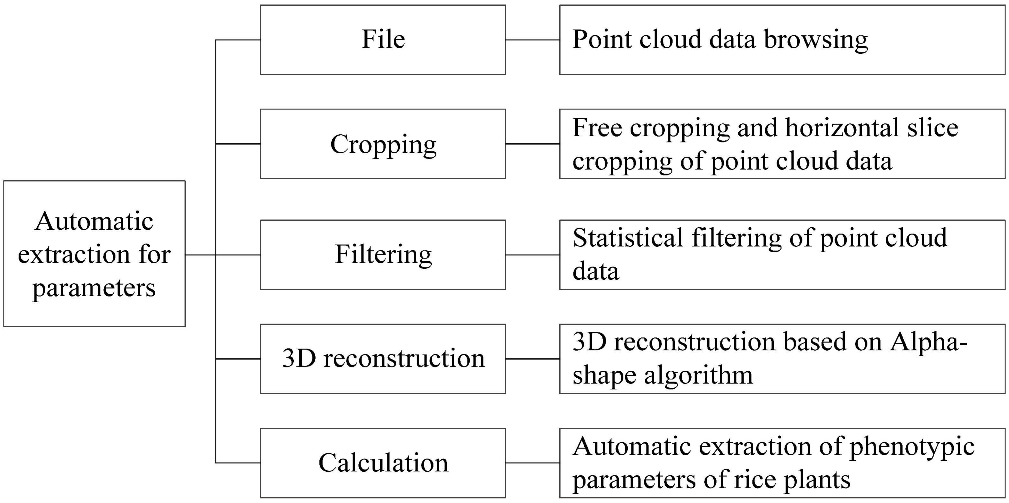
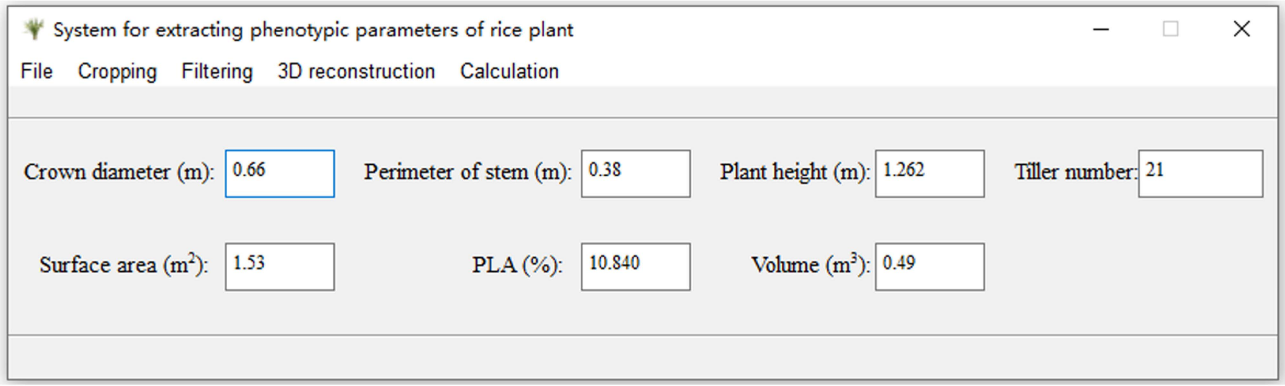

Automatic extraction for parameters
File
Point cloud data browsing
Cropping
Free cropping and horizontal slice cropping of point cloud data
Filtering
Statistical filtering of point cloud data
3D reconstruction
3D reconstruction based on Alpha-shape algorithm
Calculation
Automatic extraction of phenotypic parameters of rice plants

Figure 7. System menu function module.


System for extracting phenotypic parameters of rice plant
File Cropping Filtering 3D reconstruction Calculation

Crown diameter (m): 0.66

Perimeter of stem (m): 0.38

Plant height (m): 1.262

Tiller number: 21

Surface area (m^2): 1.53

PLA (%): 10.840

Volume (m^3): 0.49

Figure 8. GUI of the extraction tool.

5. Discussion

For a long time, due to the complexity of crop plant morphology and structure, it has been generally difficult to obtain its surface area and volume. Meanwhile, field measurement of PLA also requires professional instruments and equipment. The technology of LiDAR can effectively reflect the vertical structural characteristics of crops, which is conducive to the construction of the plant morphology and the extraction of structure parameters [26]. In this study, we developed a rice plant phenotype parameter extraction tool, and achieved the fast and efficient acquisition of rice plant phenotype parameters, even some that are difficult to obtain using traditional methods. Moreover, we selected the parameters of crown diameter, perimeter of stem, plant height, and tiller number as validation indicators to verify the accuracy, with the result of good correlation and high reliability.

Nonetheless, the feature extraction results may be influenced by the following factors. Firstly, we designed the horizontal projection centroid of the stem section as the horizontal centroid of the crown for the point cloud neighborhood search to calculate the crown diameter. Therefore, it is necessary to ensure consistency between the two centers. It is thus important to ensure the plants are placed as vertically as possible when collecting the crop plant point cloud data. Secondly, the *RRMSE* between the extracted and measured values of stem circumference in this study is slightly larger than other indicators, because it is difficult to ensure the selected stem slices are consistent with the actual observed stem parts. Finally, generally speaking, a rice plant may have 15~20 tiller branches. Because of the high number of tillers in rice plants, the rice branches and trunks are severely obstructed, and the point clouds are incomplete. Considering the aggregation and the continuity in the horizontal and vertical direction, respectively, in the stem slice point clouds, this study used the HDBSCAN algorithm to perform density segmentation and counting after the process of statistical filtering (Figure 9). This study achieved the efficient extraction of tiller numbers from the rice plant, providing assistance for rice biological breeding.

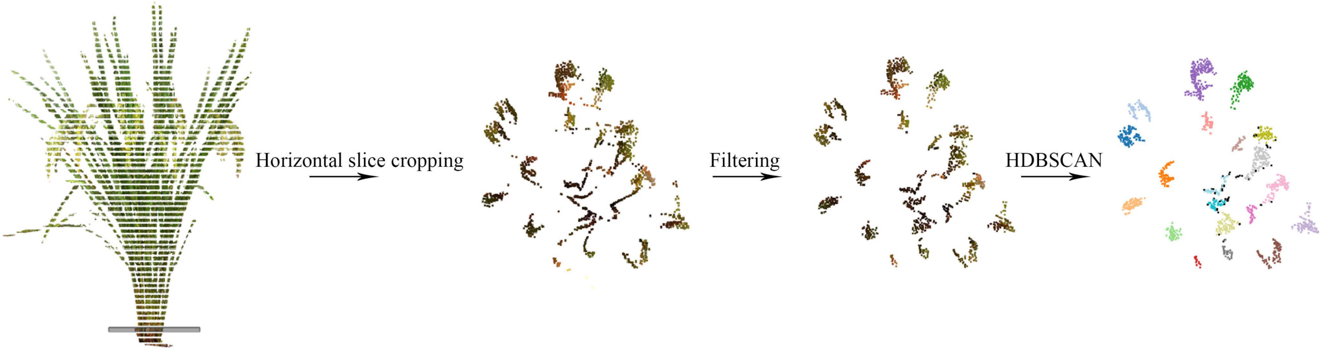


Figure 9. Extraction process of tiller number in rice plants. Different colors represent point cloud clusters on different tiller branches.

In addition, this study only used the feature extraction results of six rice plant sample point clouds for accuracy verification, with a relatively small sample size, which is also a limitation of this study. Next, we will systematically collect more plant data to carry out feature extraction and comparison verification of corresponding phenotype parameters, and further improve the accuracy of rice plant phenotype parameter extraction by designing corresponding algorithms.

6. Conclusions

In this study, we identified seven phenotypic parameters of rice plants on the basis of their characteristics, and specially proposed the methods for obtaining the tiller number. Then, a rather complete rice plant phenotype parameter extraction tool with a three-layer framework for TLS point cloud data was built based on the PyQt5 visualization framework and the Open3D library. The visual tool can achieve the automatic extraction of rice plant parameters such as crown diameter, perimeter of stem, plant height, surface area, volume, PLA, and tiller number. The results show the R^2 of crown diameter, tiller number, and plant height reached 0.80, 0.87, and 0.97, respectively, with that of perimeter of stem reaching 0.66. The *RMSE* of crown diameter, perimeter of stem and plant height was stable at the centimeter scale, and that of the tiller number was as low as 1.63. The *RRMSE* of crown diameter, plant height, and tiller number stayed within 10%, and the value of perimeter of stem was 18.29%, which indicated a high level of reliability. The tool developed in this study can achieve the goal of obtaining rice plant phenotype parameters quickly, accurately, and efficiently, which helps to improve the efficiency of rice breeding work. Although this tool is designed for rice plants, its application is not limited to rice. The extraction function of most indicators has a wide range of applicability. Additionally, the comparison and verification of phenotype feature extraction results supported by more rice plant sample data, as well as the improvement of accuracy algorithms, will be the focus of our next research.

Author Contributions: K.W. and B.L. designed the research, analyzed the data, and wrote the manuscript. K.W., B.L. and X.P. collected and analyzed the data. K.W. and X.P. performed the experiment tool. All authors have read and agreed to the published version of the manuscript.

Funding: This research work was supported by the Major Core Technology Research Project of Chongqing Academy of Agricultural Sciences under Grant No. KYLX20240500059 and Grant No. cqaas2023sjczhx003.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The scripts and datasets used and/or analyzed during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Walter, A.; Liebisch, F.; Hund, A. Plant Phenotyping: From Bean Weighing to Image Analysis. *Plant Methods* **2015**, *11*, 14. [CrossRef]
2. Weng, Y.; Zeng, R.; Wu, C.; Wang, M.; Wang, X.; Liu, Y. A Survey on Deep-Learning-Based Plant Phenotype Research in Agriculture. *Sci. Sin.-Vitae* **2019**, *49*, 698–716. [CrossRef]
3. Zhang, X.; Huang, C.; Wu, D.; Qiao, F.; Li, W.; Duan, L.; Wang, K.; Xiao, Y.; Chen, G.; Liu, Q.; et al. High-Throughput Phenotyping and QTL Mapping Reveals the Genetic Architecture of Maize Plant Growth. *Plant Physiol.* **2017**, *173*, 1554–1564. [CrossRef]
4. Bilder, R.M.; Sabb, F.W.; Cannon, T.D.; London, E.D.; Jentsch, J.D.; Parker, D.S.; Poldrack, R.A.; Evans, C.; Freimer, N.B. Phenomics: The Systematic Study of Phenotypes on a Genome-Wide Scale. *Neuroscience* **2009**, *164*, 30–42. [CrossRef]
5. Li, L.; Zhang, Q.; Huang, D. A Review of Imaging Techniques for Plant Phenotyping. *Sensors* **2014**, *14*, 20078–20111. [CrossRef]
6. Xie, W.; Wei, S.; Yang, D. Morphological Measurement for Carrot Based on Three-Dimensional Reconstruction with a ToF Sensor. *Postharvest Biol. Technol.* **2023**, *197*, 112216. [CrossRef]
7. Rosell, J.R.; Llorens, J.; Sanz, R.; Arnó, J.; Ribes-Dasi, M.; Masip, J.; Escolà, A.; Camp, F.; Solanelles, F.; Gràcia, F.; et al. Obtaining the Three-Dimensional Structure of Tree Orchards from Remote 2D Terrestrial LIDAR Scanning. *Agric. For. Meteorol.* **2009**, *149*, 1505–1515. [CrossRef]
8. Micheletto, M.J.; Chesñevar, C.I.; Santos, R. Methods and Applications of 3D Ground Crop Analysis Using LiDAR Technology: A Survey. *Sensors* **2023**, *23*, 7212. [CrossRef] [PubMed]
9. Zhang, L.; Grift, T.E. A LIDAR-Based Crop Height Measurement System for Miscanthus Giganteus. *Comput. Electron. Agric.* **2012**, *85*, 70–76. [CrossRef]
10. Sun, S.; Li, C.; Paterson, A. In-Field High-Throughput Phenotyping of Cotton Plant Height Using LiDAR. *Remote Sens.* **2017**, *9*, 377. [CrossRef]
11. Madec, S.; Baret, F.; De Solan, B.; Thomas, S.; Dutartre, D.; Jezequel, S.; Hemmerlé, M.; Colombeau, G.; Comar, A. High-Throughput Phenotyping of Plant Height: Comparing Unmanned Aerial Vehicles and Ground LiDAR Estimates. *Front. Plant Sci.* **2017**, *8*, 2002. [CrossRef] [PubMed]
12. Walter, J.D.C.; Edwards, J.; McDonald, G.; Kuchel, H. Estimating Biomass and Canopy Height With LiDAR for Field Crop Breeding. *Front. Plant Sci.* **2019**, *10*, 1145. [CrossRef] [PubMed]
13. Li, W.; Niu, Z.; Huang, N.; Wang, C.; Gao, S.; Wu, C. Airborne LiDAR Technique for Estimating Biomass Components of Maize: A Case Study in Zhangye City, Northwest China. *Ecol. Indic.* **2015**, *57*, 486–496. [CrossRef]
14. Jimenez-Berni, J.A.; Deery, D.M.; Rozas-Larraondo, P.; Condon, A.T.G.; Rebetzke, G.J.; James, R.A.; Bovill, W.D.; Furbank, R.T.; Sirault, X.R.R. High Throughput Determination of Plant Height, Ground Cover, and Above-Ground Biomass in Wheat with LiDAR. *Front. Plant Sci.* **2018**, *9*, 237. [CrossRef]
15. Nie, S.; Wang, C.; Dong, P.; Xi, X. Estimating Leaf Area Index of Maize Using Airborne Full-Waveform Lidar Data. *Remote Sens. Lett.* **2016**, *7*, 111–120. [CrossRef]
16. Liu, S.; Baret, F.; Abichou, M.; Boudon, F.; Thomas, S.; Zhao, K.; Fournier, C.; Andrieu, B.; Irfan, K.; Hemmerlé, M.; et al. Estimating Wheat Green Area Index from Ground-Based LiDAR Measurement Using a 3D Canopy Structure Model. *Agric. For. Meteorol.* **2017**, *247*, 12–20. [CrossRef]
17. Luo, S.; Wang, C.; Xi, X.; Nie, S.; Fan, X.; Chen, H.; Yang, X.; Peng, D.; Lin, Y.; Zhou, G. Combining Hyperspectral Imagery and LiDAR Pseudo-Waveform for Predicting Crop LAI, Canopy Height and above-Ground Biomass. *Ecol. Indic.* **2019**, *102*, 801–812. [CrossRef]
18. Zhou, L.; Gu, X.; Cheng, S.; Yang, G.; Shu, M.; Sun, Q. Analysis of Plant Height Changes of Lodged Maize Using UAV-LiDAR Data. *Agriculture* **2020**, *10*, 146. [CrossRef]
19. Miao, Y.; Peng, C.; Wang, L.; Qiu, R.; Li, H.; Zhang, M. Measurement Method of Maize Morphological Parameters Based on Point Cloud Image Conversion. *Comput. Electron. Agric.* **2022**, *199*, 107174. [CrossRef]
20. Shi, Y.; Wang, N.; Taylor, R.K.; Raun, W.R. Improvement of a Ground-LiDAR-Based Corn Plant Population and Spacing Measurement System. *Comput. Electron. Agric.* **2015**, *112*, 92–101. [CrossRef]
21. Guo, T.; Fang, Y.; Cheng, T.; Tian, Y.; Zhu, Y.; Chen, Q.; Qiu, X.; Yao, X. Detection of Wheat Height Using Optimized Multi-Scan Mode of LiDAR during the Entire Growth Stages. *Comput. Electron. Agric.* **2019**, *165*, 104959. [CrossRef]
22. Jin, S.; Su, Y.; Wu, F.; Pang, S.; Gao, S.; Hu, T.; Liu, J.; Guo, Q. Stem–Leaf Segmentation and Phenotypic Trait Extraction of Individual Maize Using Terrestrial LiDAR Data. *IEEE Trans. Geosci. Remote Sens.* **2019**, *57*, 1336–1346. [CrossRef]
23. Su, Y.; Wu, F.; Ao, Z.; Jin, S.; Qin, F.; Liu, B.; Pang, S.; Liu, L.; Guo, Q. Evaluating Maize Phenotype Dynamics under Drought Stress Using Terrestrial Lidar. *Plant Methods* **2019**, *15*, 11. [CrossRef]
24. Hu, F.; Lin, C.; Peng, J.; Wang, J.; Zhai, R. Rapeseed Leaf Estimation Methods at Field Scale by Using Terrestrial LiDAR Point Cloud. *Agronomy* **2022**, *12*, 2409. [CrossRef]

25. Friedli, M.; Kirchgessner, N.; Grieder, C.; Liebisch, F.; Mannale, M.; Walter, A. Terrestrial 3D Laser Scanning to Track the Increase in Canopy Height of Both Monocot and Dicot Crop Species under Field Conditions. *Plant Methods* **2016**, *12*, 9. [CrossRef]
26. Verma, M.K.; Yadav, M. Estimation of Plant's Morphological Parameters Using Terrestrial Laser Scanning-Based Three-Dimensional Point Cloud Data. *Remote Sens. Appl. Soc. Environ.* **2024**, *33*, 101137. [CrossRef]
27. Ma, X.; Wei, B.; Guan, H.; Cheng, Y.; Zhuo, Z. A Method for Calculating and Simulating Phenotype of Soybean Based on 3D Reconstruction. *Eur. J. Agron.* **2024**, *154*, 127070. [CrossRef]
28. Popescu, S.C.; Wynne, R.H.; Nelson, R.F. Measuring Individual Tree Crown Diameter with Lidar and Assessing Its Influence on Estimating Forest Volume and Biomass. *Can. J. Remote Sens.* **2003**, *29*, 564–577. [CrossRef]
29. Yan, Y.; Ding, C.; Zhang, G.; Hu, J.; Zhu, L.; Zeng, D.; Qian, Q.; Ren, D. Genetic and Environmental Control of Rice Tillering. *Crop J.* **2023**, *11*, 1287–1302. [CrossRef]
30. Bentley, J.L. Multidimensional Binary Search Trees Used for Associative Searching. *Commun. ACM* **1975**, *18*, 509–517. [CrossRef]
31. Edelsbrunner, H.; Mücke, E.P. Three-Dimensional Alpha Shapes. *ACM Trans. Graph.* **1994**, *13*, 43–72. [CrossRef]
32. Fischer, K. Introduction to Alpha Shapes. Available online: https://graphics.stanford.edu/courses/cs268-11-spring/handouts/AlphaShapes/as_fisher.pdf (accessed on 19 June 2024).
33. Campello, R.J.G.B.; Moulavi, D.; Sander, J. *Density-Based Clustering Based on Hierarchical Density Estimates*; Pei, J., Tseng, V.S., Cao, L., Motoda, H., Xu, G., Eds.; Advances in Knowledge Discovery and Data Mining; Springer: Berlin/Heidelberg, Germany, 2013; Volume 7819, pp. 160–172. [CrossRef]
34. McInnes, L.; Healy, J.; Astels, S. Hdbscan: Hierarchical Density Based Clustering. *J. Open Source Softw.* **2017**, *2*, 205. [CrossRef]
35. Timsina, J.; Humphreys, E. Performance of CERES-Rice and CERES-Wheat Models in Rice–Wheat Systems: A Review. *Agric. Syst.* **2006**, *90*, 5–31. [CrossRef]
36. Zhou, Q.-Y.; Park, J.; Koltun, V. Open3D: A Modern Library for 3D Data Processing. *arXiv* **2018**, arXiv:1801.09847. Available online: http://arxiv.org/abs/1801.09847 (accessed on 19 June 2024).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

sensors

MDPI

Article

Receptive Field Space for Point Cloud Analysis

Zhongbin Jiang, Hai Tao and Ye Liu *

School of Automation and Artificial Intelligence, Nanjing University of Posts and Telecommunications, Nanjing 210023, China; 1322059108@njupt.edu.cn (Z.J.); 1021051806@njupt.edu.cn (H.T.)
* Correspondence: yeliu@njupt.edu.cn

Abstract: Similar to convolutional neural networks for image processing, existing analysis methods for 3D point clouds often require the designation of a local neighborhood to describe the local features of the point cloud. This local neighborhood is typically manually specified, which makes it impossible for the network to dynamically adjust the receptive field's range. If the range is too large, it tends to overlook local details, and if it is too small, it cannot establish global dependencies. To address this issue, we introduce in this paper a new concept: receptive field space (RFS). With a minor computational cost, we extract features from multiple consecutive receptive field ranges to form this new receptive field space. On this basis, we further propose a receptive field space attention mechanism, enabling the network to adaptively select the most effective receptive field range from RFS, thus equipping the network with the ability to adjust granularity adaptively. Our approach achieved state-of-the-art performance in both point cloud classification, with an overall accuracy (OA) of 94.2%, and part segmentation, achieving an mIoU of 86.0%, demonstrating the effectiveness of our method.

Keywords: point cloud; receptive field; attention

1. Introduction

A point cloud is a simplified representation of an object in three-dimensional space. In recent years, with the development of technologies such as autonomous driving [1,2], 3D modeling [3], and remote sensing detection [4], point cloud analysis has become a hot topic in the field of 3D vision. It has garnered widespread attention from both the scientific and industrial communities [5–8].

The point cloud is composed of an unordered and irregular set of points ¶ $\in R^{N\times 3}$, exhibiting rotation and permutation invariance, collectively referred to as irregularity. Unlike 2D image data arranged on a pixel grid, 3D point cloud information is a continuous collection embedded in space. This makes it nearly infeasible to directly apply well-established classification and segmentation models from the image processing domain.

In this paper, we primarily focus on two typical tasks in point cloud processing: point cloud classification and segmentation. Traditional methods for solving these problems have relied on manually crafted features to capture the geometric attributes of point clouds [9,10]. Since 2017, the success of deep neural networks in image processing has sparked interest in feature learning methods for point cloud data based on various neural network architectures. Deep point cloud processing and analysis methods are rapidly advancing, surpassing traditional approaches in various tasks [11].

Due to the irregularity of point cloud data, adapting deep learning frameworks for point cloud processing is challenging. Before the emergence of PointNet [12], there were two mainstream approaches. One method involves projecting point cloud data onto different planes, and then, applying deep learning methods designed for 2D images [13–16]. The drawback of this approach is that projection can lead to occlusion and information folding, resulting in the loss of local features within the point cloud. The other method is voxel-based [17–19], which converts the original point cloud into 3D grid data, where each point corresponds to a voxel (a small cube) in the grid. Subsequent processing involves using 3D convolutions. This method's effectiveness is highly dependent on the choice of voxel size and incurs significant computational and memory overheads, making it challenging to capture detailed features.

The success of deep neural networks in image processing has spurred the development of various neural network-based methods for learning features from point cloud data. Methods for deep point cloud processing and analysis are rapidly evolving, surpassing traditional approaches in various tasks. Currently, the most mainstream deep neural networks suitable for point clouds almost all originate from PointNet [12]. PointNet designed a unique feature extractor that processes individual points at a local scale, maintaining the inherent irregularity of point clouds and directly operating on the raw point cloud data. Subsequent models such as PointNet++ [20], DGCNN [21], PointConv [22], and RSCNN [23] have largely continued PointNet's design approach. They extract point cloud information through feature extractors, and then, process the aggregated feature information using different network architectures. However, when dealing with unstructured data like point clouds, traditional convolutional neural networks often face challenges due to the irregularity and invariance in point cloud data representation. Point cloud data comprise an unordered set of points, lacking a well-defined structure like images, making traditional convolution operations difficult to apply directly. Furthermore, transformations such as rotation, translation, and scaling of point cloud data should not alter their meaning, posing additional challenges for traditional convolution operations in meeting the invariance requirements of such data.

To address these issues, researchers have begun exploring deep-learning-based methods for point cloud analysis. These methods attempt to mimic convolution operations in traditional image analysis to achieve similar feature extraction and pattern learning on point cloud data. Representative methods include set abstraction in PointNet++ [20], edge convolution in DGCNN [21], X-Conv in PointCNN [24], etc.

These methods typically simulate the size of convolution kernels by defining local neighborhoods, similar to convolution operations in CNNs. These local neighborhoods determine the range observed by the convolution operation, i.e., the receptive field. In point cloud data, the size of the receptive field directly affects the model's understanding and description of point cloud structures. Larger neighborhood ranges allow the model to capture broader contextual information, thereby achieving a more comprehensive grasp of global morphology and relationships. On the other hand, smaller neighborhood ranges are more conducive to detailed local feature extraction, enabling the model to identify and describe fine structures and patterns in point clouds more precisely. However, current methods often manually set the neighborhood size to define the receptive field range, such as setting the radius of set abstraction in PointNet++ [12] or the value of k in DGCNN [21].

To enable the network to autonomously learn and determine the range of the receptive field, we introduce a new concept called receptive field space. Receptive field space no longer views the receptive field as a single size and shape but considers it as a continuous range covering multiple scales and levels of local and global information. In receptive field space, receptive fields of different sizes and shapes are regarded as dimensions in feature space. By establishing the range of receptive fields in this dimension, the network can simultaneously consider and utilize feature information at different scales and levels. Thus, the neural network can comprehensively understand and represent the structure and relationships of the input data.

In the construction of receptive field space, we employ a simple yet efficient computational method to ensure that computational complexity does not exponentially increase with an increase in receptive field sampling density. This computational method fully utilizes the locality and sparsity of the data, effectively capturing feature information at different scales and levels while maintaining computational efficiency.

After constructing the receptive field space, we further propose a new receptive field attention mechanism called spatial-receptive field co-attention (SRCA), aiming to enable the network to adaptively focus on important receptive field ranges. The core idea of this mechanism is to dynamically adjust the receptive field range in feature space by introducing attention mechanisms, thereby achieving a more effective representation and processing of input data. The operation of the receptive field attention mechanism is similar to attention mechanisms: it allows the network to automatically adjust and allocate receptive field ranges of different sizes based on the features of the input data to focus on important areas. Specifically, by introducing attention mechanisms at different levels of the network, the importance weights of each position or feature channel can be calculated based on the current task and features of the input data. These weights can then be used to aggregate feature representations within the receptive field and generate the final output. This mechanism endows the network with the ability to autonomously adjust granularity, thereby better adapting to the feature extraction and pattern learning requirements at different scales.

This research method has been evaluated on two tasks: point cloud classification and part segmentation. The experimental results demonstrate that the method achieves state-of-the-art performance on both tasks and exhibits higher accuracy and faster inference speed compared to other methods with similar performance. Additionally, the method has relatively fewer parameters, making it more feasible and efficient for practical applications.

Our main contributions are:

- We introduce the concept of receptive field space and propose a simple and efficient method for constructing receptive field space. The introduction of receptive field space enables neural networks to learn and adjust receptive field ranges more flexibly, allowing them to adapt to features at different scales and levels.
- We propose a receptive field attention mechanism, which endows neural networks with the ability to autonomously learn and determine the receptive field range. This mechanism enables the network to dynamically adjust and focus on important receptive field ranges based on the features of the input data and task requirements, thereby improving the performance and generalization ability of the network.
- We extensively analyze and test this method, achieving state-of-the-art performance in point cloud classification and part segmentation tasks.

2. Related Works

Hand-crafted features for point clouds: Point clouds possess two main characteristics, rotation invariance and permutation invariance, collectively referred to as the unordered nature of point clouds. There is no inherent order between the arrangement of points, and swapping the order of points has no effect. Various tasks in point cloud processing and analysis, including segmentation, classification, matching, completion, and more, require constructing local features to describe the features of the point cloud. Numerous papers in computer vision and graphics have proposed local feature descriptors suitable for different problems and data structures in point clouds.

Broadly speaking, point cloud processing can be divided into intrinsic and extrinsic descriptors.

Intrinsic descriptors refer to the local characteristics of point cloud data, typically represented through relationships or statistical properties between points. For instance, the 3D shape context [25] is used to describe the local shape features of each point in a point cloud. It computes a shape context histogram for each point based on the distances and orientation relationships between point pairs in local point cloud information. Spin images [26] are used to describe the local surface shape of each point in a point cloud. They project point cloud data onto a fixed orientation grid and compute surface statistics for each direction.

Extrinsic descriptors pertain to the overall geometric and spatial properties of the point cloud data in three-dimensional space. For example, in 3D object recognition [27], they describe boundaries or boundary points in the overall point cloud to differentiate between different objects or shapes. Point feature histograms [10] describe the geometric relationships between point pairs in the overall point cloud by computing histograms of geometric features for each point pair, including normals, curvature, and relative positions between points.

These traditional methods have largely inspired numerous deep learning approaches tailored for point clouds.

Deep learning on point clouds: Methods for processing 3D point clouds using learning-based approaches can be categorized into the following types: projection-based, voxel-based, and point-based networks.

For handling irregular inputs like point clouds, an intuitive approach is to transform the irregular representation into a regular one. Given the widespread use of CNNs in 2D image processing, some methods [28–32] employ multi-view projection. In these methods, 3D point clouds are projected onto various image planes, and then, 2D CNNs extract feature representations from these image planes. Finally, multi-view feature fusion is performed to form the final output representation. However, in these methods, the geometric information within the point cloud is collapsed during the projection phase, and the sparsity of the point cloud is not well utilized. Moreover, the choice of projection planes can significantly affect recognition performance, and occlusions in three dimensions may hinder accuracy.

Another approach to processing point cloud data is 3D voxelization followed by 3D convolution using transformer-based methods [17,18,33]. However, as the number of voxels increases, the corresponding resolution grows exponentially, leading to significant computational and memory overheads. Moreover, since the point cloud is quantized into a voxel grid through certain methods, the inherent geometric details of the point cloud can be lost, resulting in a loss of accuracy. The effectiveness of voxel-based methods highly depends on the choice of voxel size. Improper voxel size selection can lead to information loss or over-sampling.

Unlike projecting or quantizing irregular point clouds onto 2D or 3D regular grids, researchers have designed deep network architectures that directly take point clouds as sets embedded in continuous space. PointNet [12] uses permutation-invariant operators like point-wise MLPs and pooling layers to aggregate features within the set. PointNet++ [20] applies these ideas within a hierarchical spatial structure to enhance sensitivity to local geometric layouts. DGCNN [21] connects the point set into a graph and performs message passing on this graph. It executes graph convolution on a KNN [34] graph, using EdgeConv to construct a local graph that generates edge features describing the relationships between points and their neighbors. Point Transformer [35] introduces the vision transformer structure suitable for 2D images, utilizing self-attention mechanisms to learn complex relationships and dependencies between points in point cloud data. This effectively integrates global and local information within point cloud data, improving feature extraction and representation learning performance. PointMlp [36] constructs an efficient pure MLP deep connectivity structure, performing feature extraction, mapping, and transformation through multiple fully connected layers.

Therefore, we believe that due to the unique complexity of point clouds, carefully designing feature extractors for local geometric structures can significantly improve the accuracy of tasks such as classification and segmentation of point clouds.

Spatial attention and channel attention: Spatial attention primarily focuses on different spatial locations within feature maps, adaptively emphasizing and adjusting the importance of this positional information. The core idea of this attention mechanism is to capture key information and important structures in images by learning the weights of each spatial location, thereby optimizing the generalization ability of the model.

On the other hand, channel attention concentrates on different channels within the feature map. It learns the weights of each channel to adjust the importance between different channels. Like spatial attention, it can also adjust the weights of individual channels through convolutional neural networks to extract key features from point clouds, improving the performance of the model in tasks related to detecting and segmenting spatially related positions.

Combining them results in the convolutional block attention module [37]. This model first applies spatial attention to adjust the importance of different spatial locations in the feature maps. Then, it applies channel attention to adjust the importance of different channels within the feature map. Finally, the feature maps adjusted by both attention modules are combined to obtain the final feature representation.

3. Method

The overall structure of the proposed method for classification and shape part segmentation is shown in Figure 1. The network is composed of several repeated units, each unit including an RFS convolutional module and an RFS attention module.

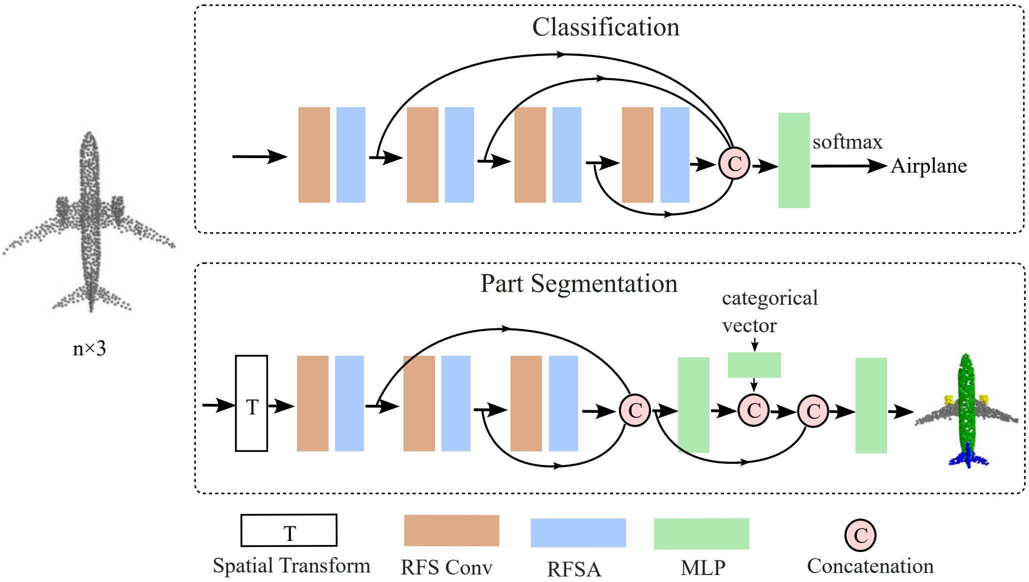


Figure 1. Overall diagram of our proposed method for two tasks: classification and shape part segmentation. RFS Conv stands for receptive field space convolution, RFSA stands for receptive field space attention, MLP stands for multi-layer perceptron.

3.1. Receptive Field Space Convolution

The receptive field (RF) refers to the specific region to which a neuron in a neural network is sensitive. In the context of convolutional neural networks (CNNs), the receptive field of a neuron denotes the area of the input image that influences the activation of that neuron. As data propagate through various layers of CNNs, the receptive field of neurons gradually increases, enabling them to capture more contextual information from the input.

In point cloud analysis, due to the non-structural and sparse nature of point cloud data, traditional convolution operations are no longer applicable. Therefore, most deep-learning-based point cloud analysis methods attempt to construct convolution operations similar to CNNs on 3D point clouds to effectively process and extract features from the data. In this process, the size and position of the receptive field define which part of the input the neuron “observes” or “perceives” to make decisions, which is crucial for understanding and pattern recognition in point cloud data.

To further enhance the processing capability of point cloud data, we introduce the concept of receptive field space. Receptive field space is formed by stacking feature maps generated by performing convolution operations on a series of different receptive field ranges.

Our definition of the stack of this series of feature maps is as follows:

$$F' = [\varphi_{r_1}(F), \varphi_{r_2}(F), \ldots, \varphi_{r_S}(F)] \tag{1}$$

where φ is some kind of basic convolution operator, and $r_1, r_2, \ldots, r_s$ is a set of increasing receptive field ranges. $[\ldots]$ is the feature stacking operation. F is the input points’ embeddings of shape $N \times D$, N is the number of points, and D is the dimension of the embedding. The output is $N \times D \times S$, with a new feature dimension: the receptive field dimension.

This construction method of receptive field space allows the network to simultaneously consider and utilize multi-scale feature information, thereby improving the ability to extract features and learn patterns from point cloud data.

EdgeConv [21] is a convolutional operation designed specifically for the characteristics of point cloud data, generating local features by utilizing the relationships between points. In this process, the concept of the receptive field plays a crucial role, determining the range that neurons can perceive when processing point cloud data. EdgeConv leverages the concept of the receptive field to enable neural networks to better understand the local structures and global relationships in point cloud data. Therefore, we adopt it as the basic convolution operator φ in our RFS convolution, which can be written as

$$\varphi_k(F) = \max_{j \in N_k(i)} ([F_i - F_j, F_i]) \tag{2}$$

Here, as shown in Figure 2, $N_k(i)$ represents the K-nearest neighbors of point i. For different choices of K, we can obtain different sizes of receptive field ranges. Therefore, we select an increasing sequence of K values $k_1, k_2, \ldots, k_s$, constructing S RFS convolution operations with different receptive field sizes, each applied to F, resulting in F′.

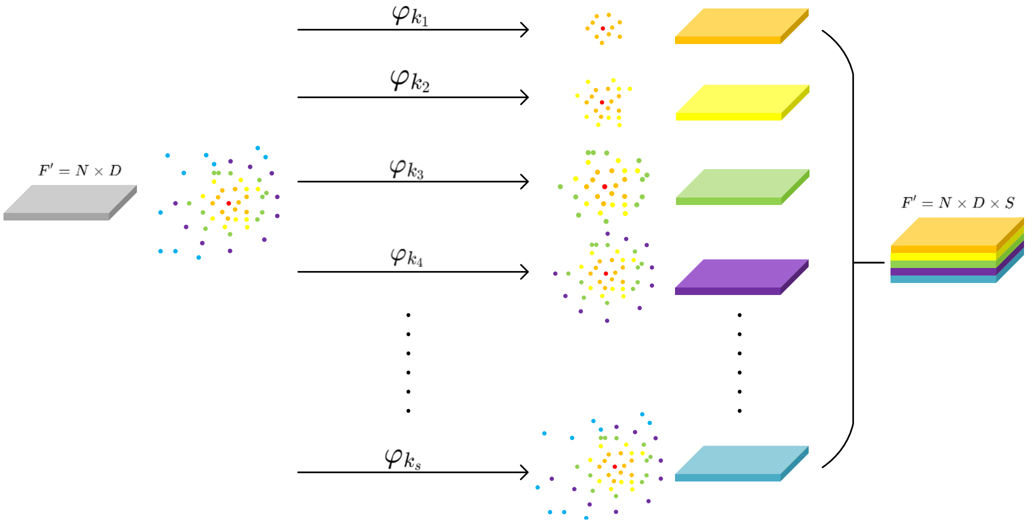


Figure 2. Aschematic diagram of RFS convolution.

3.2. Optimized Computation

RFS convolution is a novel convolution operation which involves performing the basic convolution operation φ on the input S times, which is quite time consuming. Thus, it may lead to a significant increase in computational complexity, affecting the computational efficiency of the network. Fortunately, we have discovered that there is a considerable amount of redundant computation involved in this process.

Firstly, for the K-nearest neighbors (KNN) operation, the following inclusion relationship exists:

$$N_{k_1}(i) \subset N_{k_2}(i) \subset \ldots \subset N_{k_S}(i) \tag{3}$$

In other words, we only need to compute the maximum $N_{ks}(i)$ and retain the result, then the results of other neighborhoods can be obtained in $O(1)$ time.

Next, we optimize the computation of the max operation within the neighborhood. If no optimization is performed, the time complexity of the max operation during the construction of the receptive field space is $O(k_1 + k_2 +, \ldots, +k_S)$.

We denote

$$\max_{j \in N_{k_1}(i)}([F_i - F_j, F_i]), \max_{j \in N_{k_2}(i)}([F_i - F_j, F_i]), \dots, \max_{j \in N_{k_S}(i)}([F_i - F_j, F_i]) \quad (4)$$

as

$$M_1, M_2, \dots, M_s \quad (5)$$

We partition the neighborhood into the following disjoint subsets:

$$N_{k_S}(i) = N_{k_1}(i) \cap N_{k_2}(i) - N_{k_1}(i) \cap ... \cap N_{k_S}(i) - N_{k_{S-1}}(i) \quad (6)$$

Within each subset, perform the max operation to obtain a series of maximum values $m_1, m_2, \dots, m_s$; the time complexity here is $O(k_s)$. Then, the calculations $M_1 = m_1$, $M_2 = \max(M_1, m_1), \dots, M_S = \max(M_{S-1}, m_S)$ can be carried out successively. The time complexity here is $O(S)$, and the overall time complexity is reduced to $O(k_s + S)$.

By optimizing the computation process of constructing the receptive field space, specifically the RFS convolution process, we can improve the network's computational efficiency without compromising the accuracy of the receptive field calculation.

3.3. Attention for Receptive Field Space

In the preceding sections, we introduced the RFS convolution operation and the optimized receptive field calculation method for point cloud data processing. Next, we will delve into the design and application of the receptive field attention mechanism.

By computing, we have constructed the receptive field space. We then design a receptive-wise attention mechanism to operate on this new feature dimension, allowing the network to adaptively focus on important receptive field ranges, thereby achieving self-adjustment of feature extraction granularity.

Here, we construct two types of attention: (1) channel-wise attention (CA) and (2) spatial-receptive field co-attention (SRCA). The schematic diagram of the two attentions is shown in Figure 3.

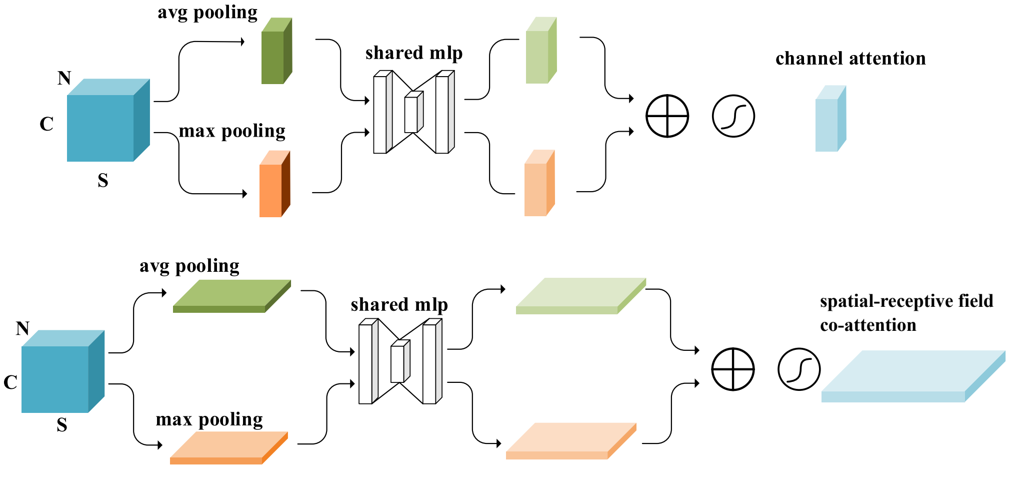


Figure 3. A schematic diagram of channel-wise attention and spatial-receptive field co-attention.

First, let us examine channel-wise attention (CA). Channel-wise attention primarily focuses on the relationships between different channels in the feature map. In our design, we first utilize average pooling and max pooling operations to compress the spatial and receptive field joint dimensions of the feature map, reducing redundant information and computational load. Then, we use a multi-layer perceptron (MLP) to integrate the compressed features and generate the channel attention weights.

Spatial-receptive field co-attention (SRCA) goes a step further by considering both spatial information and the receptive field range. In this design, we address not only the relationships between the channels in the feature map but also their relationships in the spatial and receptive field dimensions. Therefore, we combine spatial and receptive field information to design a co-attention mechanism. Specifically, we also use average pooling and max pooling to compress the channel dimension of the feature map, and then, utilize an MLP to generate the co-attention weights.

In constructing the overall attention mechanism, we drew inspiration from the CBAM (convolutional block attention module). The traditional spatial attention module CBAM was initially applied in image processing, and thus, includes two spatial dimensions. Following the CBAM approach, we implemented this traditional spatial attention module in point cloud processing, using CA+SA (channel attention and spatial attention) to achieve point cloud spatial attention. The feature maps used by these methods are derived from point cloud data processed through KNN (k-nearest neighbors). While effective, the performance was average.

Therefore, we changed our approach. We first extracted spatial features in layers within the receptive field space (RFS), and then, used CA and SRCA (spatial-receptive field co-attention) to process the data. This method allowed us to consider both spatial and receptive field dimensions simultaneously, thereby improving point cloud processing effectiveness.

RFS extracts spatial features in layers, enabling the model to better capture local and global features in the point cloud. After extracting features using RFS, we further applied CA and SRCA to enhance the feature representation. CA captures the importance of different channels in the point cloud, while SRCA provides a more precise attention mechanism by combining spatial dimensions and receptive field features.

Since images have two spatial dimensions, while our feature maps have one spatial dimension and one receptive field dimension, in form, our CA and SRCA are consistent with the channel-wise attention and spatial attention in CBAM. The specific expression is as follows:

$$CA = \psi_{CA}(Avp_{sr}(F')) + \psi_{CA}(Map_{sr}(F')) \tag{7}$$

$$SRCA = \psi_{SRCA}(Avp_{ch}(F')) + \psi_{SRCA}(Map_{ch}(F')) \tag{8}$$

where Avp_{ch} and Map_{ch} represent average and max pooling along the feature channel dimension, respectively. ψ_{SRCA} denotes the MLP for SRCA attention. Avp_{sr} and Map_{sr} are average and max pooling performed jointly along the spatial and receptive field dimensions, respectively. ψ_{CA} represents the MLP for CA.

Although formally these two attentions resemble CBAM, their content is indeed different. In CBAM, images have two spatial dimensions, while point clouds only have one spatial dimension, and we constructed a new receptive field dimension in point clouds. This enables our attention mechanism not only to focus the network on important feature channels and spatial positions but also to pay attention to significant receptive field ranges, thereby achieving adaptive adjustment of feature extraction granularity.

4. Experiments

4.1. Classification

We evaluate our classification model on the ModelNet40 [18] dataset, which comprises 12,311 CAD models meshed from 40 categories, including 9843 training samples and 2468 test samples. Each point cloud instance is uniformly sampled with 1024 points, utilizing only the spatial coordinates (x, y, z) for classification. Prior to training, we augment the point cloud data through random rotation and scaling.

The network structure used for the classification task is depicted in the upper part of Figure 1. In classification tasks, SRCA aggregates local features into global features through pooling operations for the classification of the entire point cloud. In segmentation tasks, SRCA not only utilizes local features but also propagates global features back to each point, allowing each point to have both local information and global context for refined segmentation. We employ a combination of four RFS convolution along with receptive-wise attention to adaptively extract and enhance the required spatial feature information across scale space. These four layers share a common fully connected layer and independently compute spatial feature information. For all RFS convolutions involved in the classification task, the value of k in the KNN is set to 5, 10, 15, and 20. After concatenating the four sets of high-dimensional spatial feature information, they are fed into a shared fully connected layer with 1024 dimensions. Subsequently, maxpool and avgpool are used to obtain all global features of the current point cloud information. Multiple MLPs are then used to transform these global features, followed by sigmoid normalization. We retrain the model on the entire training dataset and evaluate it on the test dataset.

We use SGD with a learning rate of 0.1 and employ cosine annealing [38] to reduce the learning rate to 0.001. The batch size is set to 16 with a momentum of 0.9.

Following the mainstream evaluation criteria for this dataset, we primarily focus on overall accuracy (OA) to compare our results with those of other models. We also compared the model's inference speed and parameter volume. The OA (overall accuracy) metric is commonly used to evaluate the overall performance of a model. Parameter count reflects the complexity of the model, where more parameters indicate higher computational resource consumption and increased risk of overfitting. Speed typically reflects the efficiency of the model during the inference phase, with faster inference speed being crucial for real-time applications or tasks requiring large-scale data processing. The models we compare against include PointNet [12], PointNet++ [20], PointCNN [24], DGCNN [21], Point Transformer [35], CurveNet [39], PointNeXt [40], and PointMLP [36], We conducted tests using these open-source models.

The comparison results are shown in Table 1. Our model achieved excellent results on the classification task of this dataset, with an overall accuracy (OA) reaching 94.2%, comparable to the state of the art. Our model also performs well in terms of parameters and speed. Compared to PointMLP, our model outperforms it in both parameters and speed.

Table 1. Classification on the ModelNet40 dataset.

Method	OA (%)	Speed (ins./s)	Params (M)
PointNet [12]	89.2	2283	3.48
PointNet++ [20]	90.7	1268	1.48
PointCNN [24]	92.5	152	8.20
DGCNN [21]	92.9	535	1.80
PointTrans. [35]	93.7	90	2.94
CurveNet [39]	93.8	145	2.03
PointNeXt [40]	93.2	760	1.51
PointMLP [36]	94.2	120	12.36
Our method	94.2	476	1.82

4.2. More Experiments on ModelNet40

4.2.1. Ablation Experiments

This experiment aims to evaluate the impact of our method on the classification performance of the model. By gradually adding, removing, or altering certain layers in the network, we explore their influence on the overall performance of the model and verify their effectiveness in improving the key indicator OA (overall accuracy) in the classification task.

We conducted the experiment from two perspectives. In the first perspective, we modified the receptive-wise attention mechanism (receptive-wise attention) in Figure 2, replacing CA and SRCA with pure MLP structures, self-attention structures, and the CBAM we designed for point clouds. The first two structures are commonly used in mainstream point cloud segmentation methods, such as Point Transformer, CurveNet, and PointMLP, to enhance the extracted point cloud feature information. The latter method, CBAM, is widely used in image processing, and we designed a point cloud version to validate the effectiveness of SRCA.

In the second aspect of the experiment, we intended to test the impact of different depths of receptive-wise attention on the overall results, so we did not use all of receptive-wise attention to enhance feature information. We experimented by retaining n receptive-wise attention layers in order from deep to shallow. For example, using three receptive-wise attention layers means retaining the last three receptive-wise attention layers in the network and discarding the shallowest one. For experimental rigor, the RFS convolution preceding receptive-wise attention remains unchanged in both experiments. In all experiments, the information outputted by RFS convolution is set to the same adaptive multi-scale spatial features.

The experimental results are shown in Table 2. From the perspective of enhancing feature information, our receptive-wise attention mechanism, which uses CA+SRCA, significantly outperforms other point cloud classification methods such as multi-layer MLP and direct self-attention, as well as the CBAM we designed for point clouds. In terms of the effectiveness of dynamic adjustment at different levels, our receptive-wise attention mechanism is evidently suitable for all levels in the network; otherwise, the classification performance would decline.

Table 2. Ablation experiments: the "CBAM" was implemented by us in point cloud processing tasks based on our understanding of the CBAM method from image processing.

Number of Receptive-Wise Attention	MLP	CA+SRCA	Self-Attention	CBAM	OA (%)
4				✓	93.5
4			✓		93.9
4	✓				93.7
4		✓			94.2
3		✓			93.7
2		✓			93.3
1		✓			92.9

4.2.2. Feature Points

Feature points are those particularly highlighted by receptive fields of varying sizes within the receptive field space. In point cloud classification tasks, these feature points play a crucial role, representing the local features of the point cloud data, which include geometric information and the local structure of the object surface. RFS convolution extracts feature points based on different k values. By accurately extracting and describing these feature points, we can transform point cloud data into distinctive feature representations, providing essential input for classification models. The selection and description of feature points directly affect the model's performance and accuracy, helping the model accurately recognize and classify different objects or scenes, thereby achieving more precise point cloud classification tasks.

The introduction of receptive field space brings a new perspective to point cloud classification tasks. By constructing a series of receptive fields of different sizes to form a new feature dimension, the receptive field space allows the model to observe point cloud data comprehensively, thereby better understanding the data's structure and features, as shown in Figure 4. This image illustrates the results of spatial feature point extraction by the receptive field space (RFS) and attention for receptive field space. The leftmost part shows the input point cloud data, followed by the spatial feature points extracted by the model under different receptive field sizes. These pieces of information are merged, and then, the attention for receptive field space adaptively adjusts the weights of the receptive field features, resulting in the output of adjusted spatial feature points. Clearly, our receptive field space can cover nearly the entire model and extract spatial feature information, which is further refined by channel-wise attention and spatial-receptive field co-attention for adaptive adjustment.

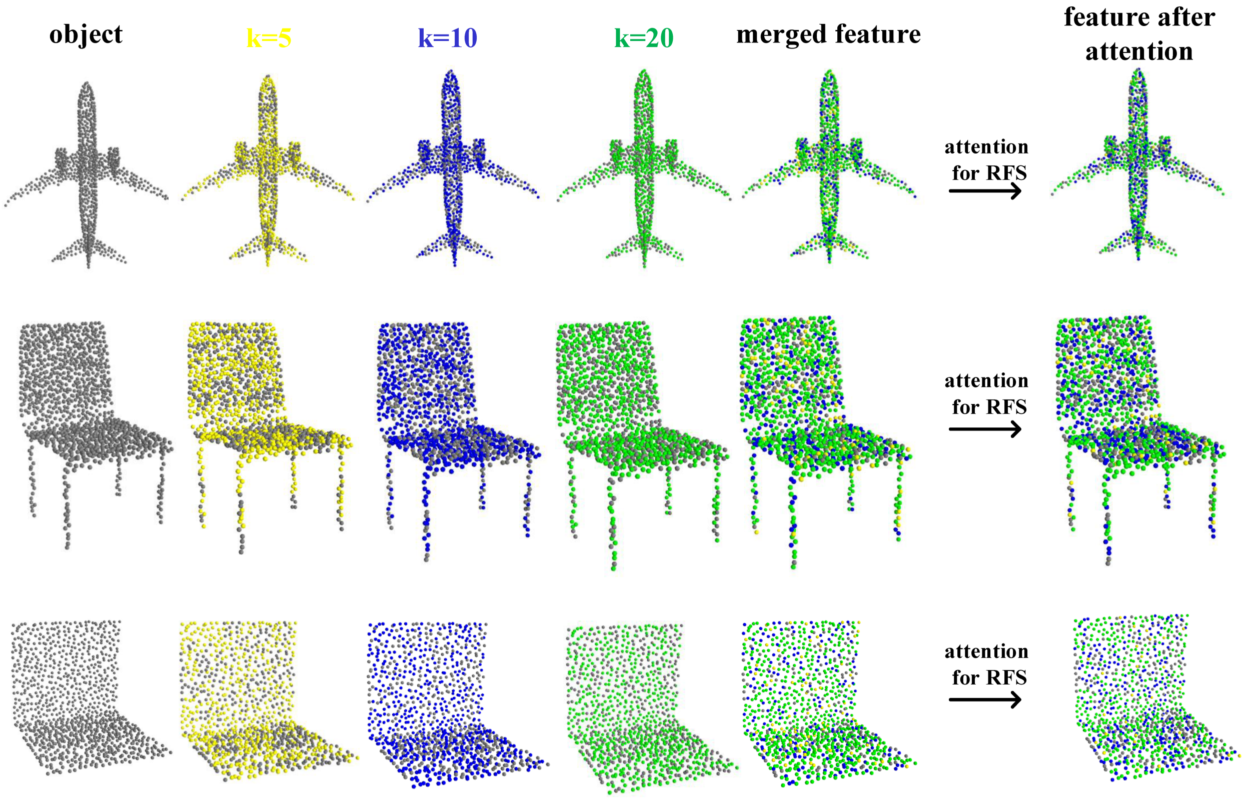


Figure 4. Visualization of feature point extraction and processing.

Simultaneously, the receptive field attention mechanism further optimizes the feature point extraction process. By allowing the model to adaptively focus on receptive field ranges with significant information, the receptive field attention mechanism enhances the model's ability to capture and identify key points. This mechanism enables the model to focus more on crucial local feature points in classification tasks, thereby further improving the performance and accuracy of the classification model.

4.2.3. Model's Versatility

For point cloud classification tasks, the versatility of a model implies its ability to adapt to point cloud data with different quantities and densities, yielding satisfactory classification results across various scenarios.

Based on this concept, we conducted a series of experiments to validate the versatility of our point cloud classification model. This experimental design fully considered the diversity of point cloud data in real-world scenarios and evaluated the model's performance from different perspectives. As shown in Figure 5, we trained the model using different numbers of input points (1024, 512, 256, and 128 points) and observed its performance in point cloud classification tasks under these varying input conditions. Regardless of whether the point cloud density was high or low, our model consistently achieved good classification results, confirming its excellent versatility.

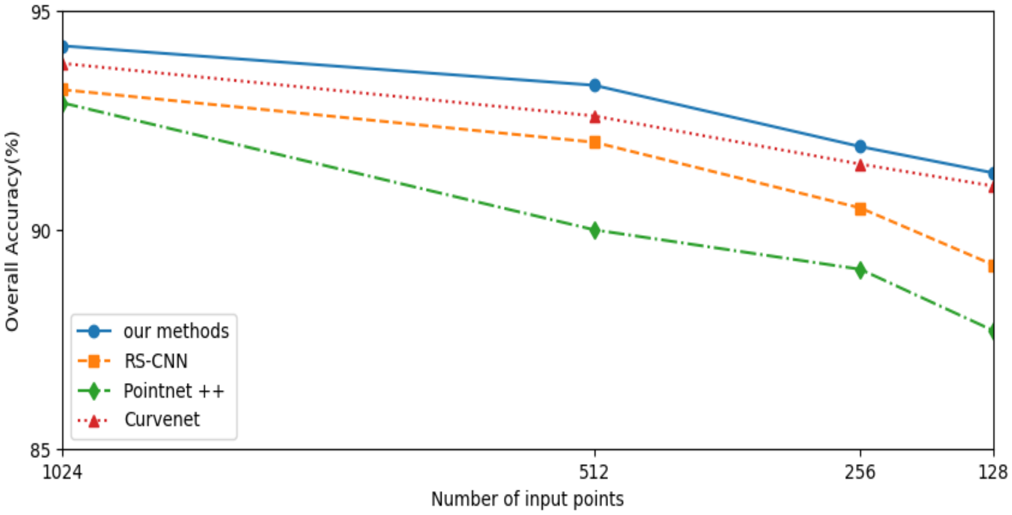


Figure 5. Comparison on sparser training and testing input point cloud.

4.2.4. Receptive Field Space Visualization

Our model uses channel-wise attention and spatial-receptive field co-attention to process the features extracted by RFS convolution. This results in varying receptive field intensity representations for each point, highlighting the importance of different receptive field ranges and enabling the model to adaptively focus on key features at different scales.

As shown in Figure 6, our model achieves adaptive receptive field adjustment by representing the intensity of receptive field information for each point in high-dimensional space. In smoother regions, the receptive field is relatively larger because these areas require more contextual information to accurately describe their overall shape. In contrast, in regions rich in details, the receptive field is relatively smaller to capture local structures and fine variations more precisely. Consequently, our model can comprehensively understand the local structures and features at different scales within point cloud data, providing more accurate and comprehensive feature representations for subsequent tasks.

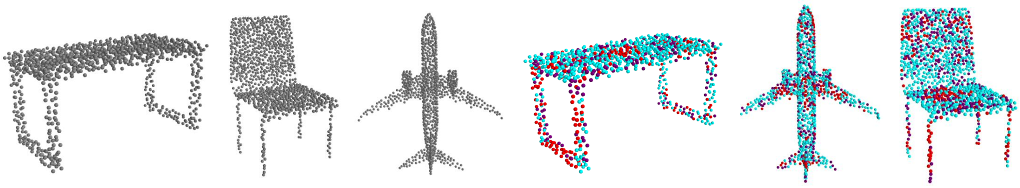


Figure 6. Each point in the point cloud is represented with different receptive field information.

4.3. Part Segmentation

We evaluate our segmentation model on the ShapeNetPart dataset, which contains 16,881 shapes from 16 categories, annotated with a total of 50 parts. Most object categories are labeled with two to five parts. We use the average intersection over union (IoU) across all instances as the evaluation metric. For each point cloud instance, it is uniformly sampled with 2048 points.

The network architecture is shown in the lower part of Figure 1. After passing through the spatial transformation network, we utilize a combination of three RFS convolutions and receptive-wise attention to extract spatial feature information. A shared fully connected layer (1024) aggregates information from the preceding layers. Next, we append label information that has been embedded. Finally, three shared fully connected layers (256, 256, 128) are used to transform point features.

We also use SGD with a learning rate of 0.1 and employ cosine annealing to reduce the learning rate to 0.001. The batch size is set to 32 with a momentum of 0.9.

The final segmentation results are shown in Table 3, where we compare our results with PointNet [12], PointNet++ [20], DGCNN [21], SPLATNet [41], SpiderCNN [42], PointNeXt [40], and PointMLP [36]. While our segmentation results are not the best, they surpass the 86.0% threshold and rank among the top performers.

Table 3. Part segmentation results on the ShapeNetPart dataset.

Method	Inst.mIoU (%)
PointNet [12]	83.7
PointNet++ [20]	85.1
DGCNN [21]	85.2
SPLATNet [41]	85.4
SpiderCNN [42]	85.3
PointMLP [36]	86.1
PointNeXt [40]	86.3
Our method	86.0

The results in Table 4 show the segmentation outcomes of our model on all instances in the ShapeNetPart dataset. Our model achieves excellent segmentation results for categories that emphasize spatial features, such as chair, airplane, skateboard, and table.

Table 4. The segmentation results for all instances in the ShapeNetPart dataset. Red, blue, and green, respectively, represent the first, second, and third rankings of the data.

Method	Inst. mIoU	plane	bag	cap	car	chair	ear phone	guitar	knife	lamp	laptop	motor	mug	pistol	rocket	skate board	table
PointNet [12]	83.7	83.4	78.7	82.5	74.9	89.6	73.0	91.5	85.9	80.8	95.3	65.2	93.0	81.2	57.9	72.8	80.6
PointNet++ [20]	85.1	82.4	79.0	87.7	77.3	90.8	71.8	91.0	85.9	83.7	95.3	71.6	94.1	81.3	58.7	77.4	82.6
DGCNN [21]	85.2	84.0	83.4	86.7	77.8	90.6	74.7	91.2	87.5	82.8	95.7	66.3	94.9	81.1	63.5	74.5	82.6
SPLATNet [41]	85.4	83.2	84.3	89.1	80.3	90.7	75.5	92.1	87.1	82.6	96.1	75.6	95.2	83.8	64.0	75.5	81.8
SpiderCNN [42]	85.3	83.5	81.0	87.2	77.5	90.7	76.8	91.1	87.3	83.3	95.8	70.2	93.5	82.7	59.7	75.8	82.8
PointMLP [36]	86.1	83.5	83.4	87.5	80.5	90.3	78.2	92.2	88.1	83.9	96.2	75.2	95.8	85.4	64.6	83.3	84.3
PointNeXt [40]	86.3	83.9	83.6	86.2	81.1	90.5	77.3	91.5	88.4	82.3	95.9	77.5	95.6	84.4	66.1	83.5	84.4
Our Method	86.0	84.8	84.2	87.3	79.1	90.6	74.7	90.9	88.2	83.0	95.8	67.4	93.5	82.3	64.5	83.1	84.6

4.4. Optimized Computation Experiment

We conducted an optimized computation experiment using an NVIDIA RTX 3090 GPU, primarily aimed at testing the speed performance of the model during training and inference. To evaluate the effectiveness of the optimized computation method, we kept the experimental setup consistent with the previous classification and segmentation experiments, and set the parameter S in the receptive field space (RFS) to 4, using four spatial features for stacking in the experiment.

For the evaluation of the classification task, we continued to use the ModelNet40 [18] dataset, a widely used benchmark dataset for 3D object classification. For the evaluation of the segmentation task, we still used the ShapeNetPart [43] dataset, commonly used for 3D object part segmentation. We tested the results of both tasks before and after applying the optimized computation method.

As shown in Table 5, our optimized computation method with S = 4 demonstrated a significant improvement compared to the non-optimized version. The optimized method not only showed a noticeable increase in training speed but also exhibited a substantial advantage in inference speed. Specifically, in the classification task, the optimized model showed significant improvements in computational efficiency. In the segmentation task, the optimized model also demonstrated considerable improvements in processing time.

Table 5. Optimized computation experiment results.

	Optimized Computation Speed (ins./s)	Non-Optimized Computation Speed (ins./s)
Classification Training	185	67
Classification Inference	476	243
Segmentation Training	78	39
Segmentation Inference	245	134

4.5. Visualization of Segmentation Results

In this section, we will detail the model's segmentation process and visualize its segmentation results.

Figure 7 illustrates the visualization of the model's segmentation process. Feature points from different parts are aggregated through multiple network layers to gather high-dimensional spatial feature information, ultimately focusing on specific parts. This process demonstrates the model's refinement and optimization in the segmentation task.

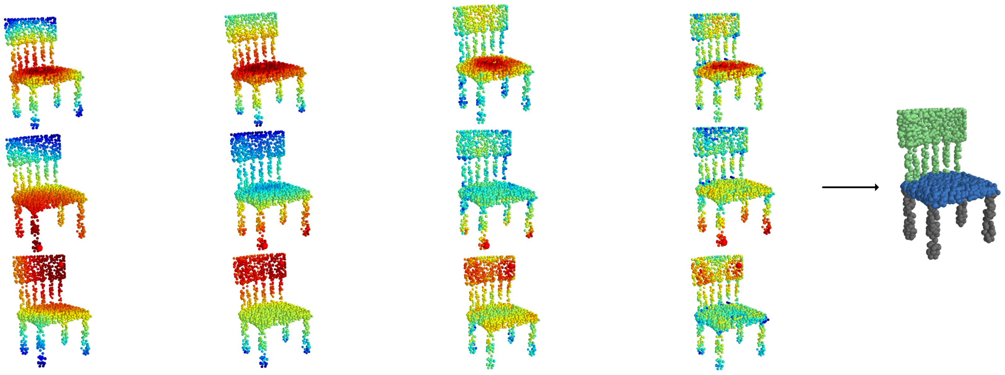


Figure 7. Visualization of segmentation process.

Figure 8 shows the final segmentation results. On the left are the segmentation results of DGCNN, a model that also utilizes spatial information. In the middle are the ground truth labels from the dataset, and on the right are the segmentation results obtained using our method. A clear comparison shows that our method outperforms others in terms of segmentation quality, exhibiting higher accuracy and reliability.

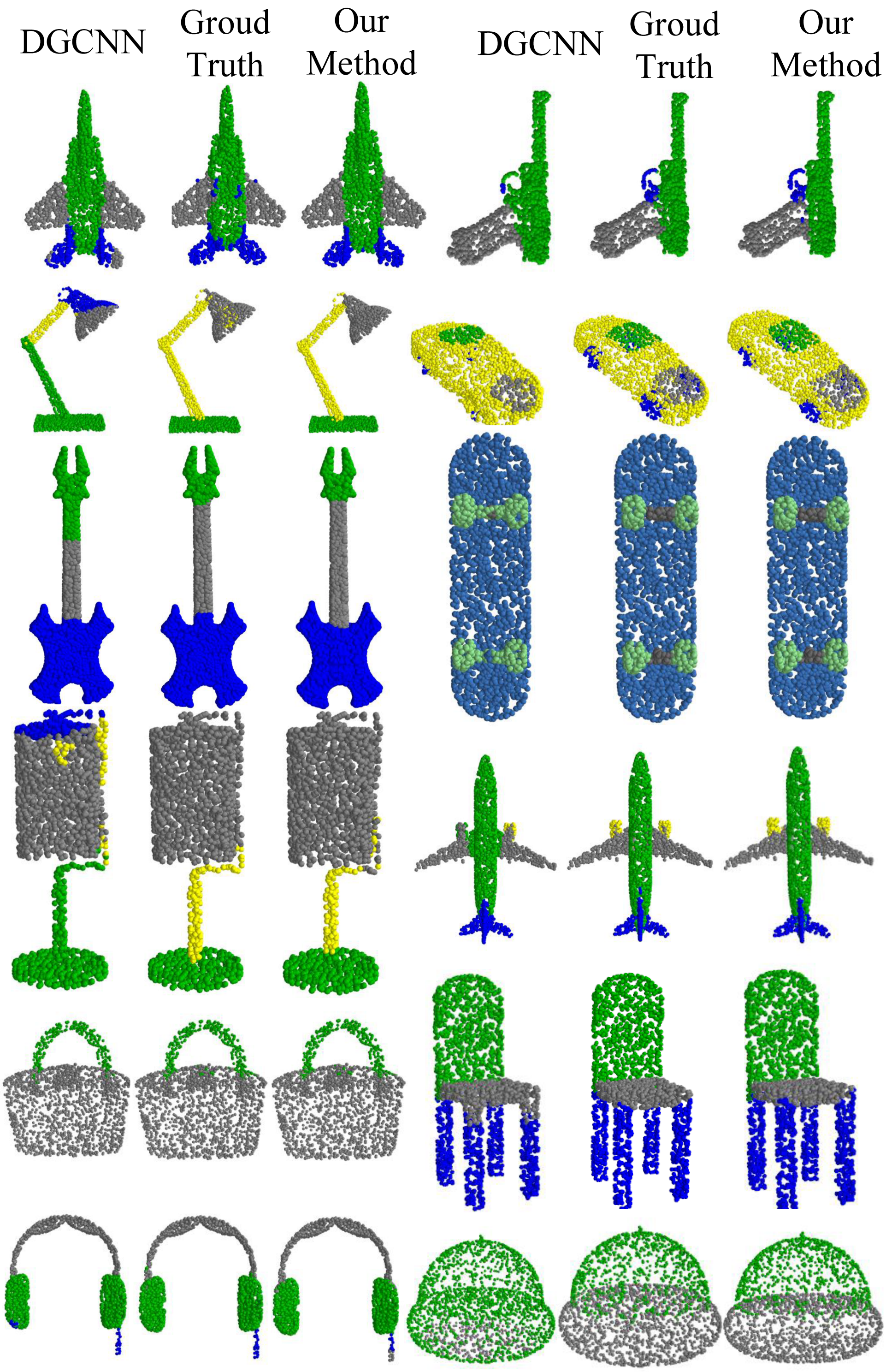


Figure 8. Visualization of segmentation results.

5. Discussion

In this work, we explored the receptive field space and proposed the receptive field attention mechanism based on it. We designed experiments to demonstrate the performance of this method in mainstream point cloud classification and segmentation tasks. Our experiments validated the feasibility of allowing the network to autonomously learn and determine the receptive field range. Our approach is not limited to traditional point cloud feature extraction methods but achieves more accurate and comprehensive capture of spatial local feature points by introducing receptive field space and the receptive field attention mechanism. This method enables the model to better understand the structure and features of point cloud data, thereby improving the effectiveness of classification and segmentation tasks.

By employing carefully designed feature extractors to extract and enhance features, we can more effectively accomplish classification and segmentation tasks. Through the introduction of receptive field space and the receptive field attention mechanism, we further optimize the feature extraction process, allowing the model to better focus on important receptive field ranges, thereby enhancing its ability to capture and identify key points.

However, we also recognize the challenges facing current point cloud classification and segmentation tasks. Firstly, the inherent noise and uncertainty in datasets make it difficult for models to learn and generalize accurately. Future improvements may require more complex representational capabilities to capture this diversity. Secondly, mainstream models have already identified optimal model structures and parameter settings for these datasets, and further improvements may require more domain-specific knowledge or innovative technologies. Finally, although our method can adaptively adjust the receptive field range to achieve autonomous granularity adjustment, the feature extraction method we use still belongs to traditional methods. We hope to innovate the extraction method in our future research.

Author Contributions: Conceptualization, H.T. and Z.J.; methodology, Z.J. and H.T.; software, H.T. and Z.J.; validation, Z.J. and H.T.; formal analysis, Z.J. and H.T.; investigation, Z.J. and H.T.; resources, Z.J., H.T. and Y.L.; data curation, Z.J. and H.T.; writing—original draft preparation, Z.J. and Y.L.; writing—review and editing, Z.J.; visualization, Z.J. All authors have read and agreed to the published version of the manuscript.

Funding: The research work of this paper is sponsored by Natural Science Foundation for Colleges and Universities in Jiangsu Province No. 21KJB520015, and sponsored by Jiangsu Postdoctoral Fund grand No. 2021K398C.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Yigzaw, S. An Analysis and Benchmarking in Autoware. AI and OpenPCDet LiDAR-Based 3D Object Detection Models. Master's Thesis, University of Waterloo, Waterloo, ON, Canada, 2023.
2. Shi, S.; Wang, X.; Li, H. Pointrcnn: 3d object proposal generation and detection from point cloud. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–17 June 2019; pp. 770–779.
3. Liu, W.; Sun, J.; Li, W.; Hu, T.; Wang, P. Deep learning on point clouds and its application: A survey. *Sensors* **2019**, *19*, 4188. [CrossRef] [PubMed]
4. Stilla, U.; Xu, Y. Change detection of urban objects using 3D point clouds: A review. *Isprs J. Photogramm. Remote Sens.* **2023**, *197*, 228–255. [CrossRef]
5. Park, J.; Kim, C.; Kim, S.; Jo, K. PCSCNet: Fast 3D semantic segmentation of LiDAR point cloud for autonomous car using point convolution and sparse convolution network. *Expert Syst. Appl.* **2023**, *212*, 118815. [CrossRef]
6. Xie, T.; Wang, L.; Wang, K.; Li, R.; Zhang, X.; Zhang, H.; Yang, L.; Liu, H.; Li, J. FARP-Net: Local-global feature aggregation and relation-aware proposals for 3D object detection. *IEEE Trans. Multimed.* **2023**, *26*, 1027–1040. [CrossRef]
7. Shi, S.; Guo, C.; Jiang, L.; Wang, Z.; Shi, J.; Wang, X.; Li, H. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 10529–10538.
8. Chen, X.; Shi, S.; Zhu, B.; Cheung, K.C.; Xu, H.; Li, H. Mppnet: Multi-frame feature intertwining with proxy points for 3d temporal object detection. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2022; Springer: Cham, Switzerland, 2022; pp. 680–697.
9. Lu, M.; Guo, Y.; Zhang, J.; Ma, Y.; Lei, Y. Recognizing objects in 3D point clouds with multi-scale local features. *Sensors* **2014**, *14*, 24156–24173. [CrossRef] [PubMed]

10. Rusu, R.B.; Blodow, N.; Beetz, M. Fast point feature histograms (FPFH) for 3D registration. In Proceedings of the 2009 IEEE iNternational Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; IEEE: New York, NY, USA, 2009; pp. 3212–3217.
11. Chang, A.X.; Funkhouser, T.; Guibas, L.; Hanrahan, P.; Huang, Q.; Li, Z.; Savarese, S.; Savva, M.; Song, S.; Su, H.; et al. Shapenet: An information-rich 3d model repository. *arXiv* **2015**, arXiv:1512.03012.
12. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660.
13. Yang, Q.; Chen, H.; Ma, Z.; Xu, Y.; Tang, R.; Sun, J. Predicting the perceptual quality of point cloud: A 3d-to-2d projection-based exploration. *IEEE Trans. Multimed.* **2020**, *23*, 3877–3891. [CrossRef]
14. Navaneet, K.; Mandikal, P.; Agarwal, M.; Babu, R.V. Capnet: Continuous approximation projection for 3d point cloud reconstruction using 2d supervision. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 8819–8826.
15. Remondino, F. From point cloud to surface: The modeling and visualization problem. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2003**, *34*, W10.
16. Jovančević, I.; Pham, H.H.; Orteu, J.J.; Gilblas, R.; Harvent, J.; Maurice, X.; Brèthes, L. 3D point cloud analysis for detection and characterization of defects on airplane exterior surface. *J. Nondestruct. Eval.* **2017**, *36*, 74. [CrossRef]
17. Maturana, D.; Scherer, S. Voxnet: A 3d convolutional neural network for real-time object recognition. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–3 October 2015; IEEE: New York, NY, USA, 2015; pp. 922–928.
18. Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; Xiao, J. 3d shapenets: A deep representation for volumetric shapes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1912–1920.
19. Li, G.; Müller, M.; Thabet, A.; Ghanem, B. DeepGCNs: Can GCNs Go as Deep as CNNs? In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019.
20. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In Proceedings of the Advances in Neural Information Processing Systems 30 (NIPS 2017): 31st Annual Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017.
21. Simonovsky, M.; Komodakis, N. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 3693–3702.
22. Wu, W.; Qi, Z.; Fuxin, L. Pointconv: Deep convolutional networks on 3d point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–17 June 2019; pp. 9621–9630.
23. Liu, Y.; Fan, B.; Xiang, S.; Pan, C. Relation-shape convolutional neural network for point cloud analysis. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–17 June 2019; pp. 8895–8904.
24. Li, Y.; Bu, R.; Sun, M.; Wu, W.; Di, X.; Chen, B. Pointcnn: Convolution on x-transformed points. In Proceedings of the Advances in Neural Information Processing Systems 31 (NIPS 2018): 32nd Annual Conference on Neural Information Processing Systems, Montreal, QC, Canada, 3–8 December 2018.
25. Körtgen, M.; Park, G.J.; Novotni, M.; Klein, R. 3D shape matching with 3D shape contexts. In Proceedings of the 7th Central European Seminar on Computer Graphics, Budmerice Slovakia, 22–24 April 2003; Volume 3, pp. 5–17.
26. Johnson, A.E. Spin-Images: A Representation for 3-D Surface Matching. Ph.D. Thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 1997.
27. Guo, Y.; Bennamoun, M.; Sohel, F.; Lu, M.; Wan, J. 3D object recognition in cluttered scenes with local surface features: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 2270–2287. [CrossRef] [PubMed]
28. Chen, X.; Ma, H.; Wan, J.; Li, B.; Xia, T. Multi-view 3d object detection network for autonomous driving. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1907–1915.
29. Kanezaki, A.; Matsushita, Y.; Nishida, Y. Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 5010–5019.
30. Lang, A.H.; Vora, S.; Caesar, H.; Zhou, L.; Yang, J.; Beijbom, O. Pointpillars: Fast encoders for object detection from point clouds. In Proceedings of the IEEE/CVF Conference On Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–17 June 2019; pp. 12697–12705.
31. Li, B.; Zhang, T.; Xia, T. Vehicle detection from 3d lidar using fully convolutional network. *arXiv* **2016**, arXiv:1608.07916.
32. Su, H.; Maji, S.; Kalogerakis, E.; Learned-Miller, E. Multi-view convolutional neural networks for 3d shape recognition. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 945–953.
33. Song, S.; Yu, F.; Zeng, A.; Chang, A.X.; Savva, M.; Funkhouser, T. Semantic scene completion from a single depth image. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1746–1754.
34. Guo, G.; Wang, H.; Bell, D.; Bi, Y.; Greer, K. KNN model-based approach in classification. In Proceedings of the On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS,

DOA, and ODBASE 2003, Catania, Sicily, Italy, 3–7 November 2003; Proceedings; Springer: Berlin/Heidelberg, Germany, 2003; pp. 986–996.

35. Zhao, H.; Jiang, L.; Jia, J.; Torr, P.H.; Koltun, V. Point transformer. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 16259–16268.
36. Ma, X.; Qin, C.; You, H.; Ran, H.; Fu, Y. Rethinking network design and local geometry in point cloud: A simple residual MLP framework. *arXiv* **2022**, arXiv:2202.07123.
37. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.
38. Loshchilov, I.; Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv* **2016**, arXiv:1608.03983.
39. Muzahid, A.; Wan, W.; Sohel, F.; Wu, L.; Hou, L. CurveNet: Curvature-based multitask learning deep networks for 3D object recognition. *IEEE/CAA J. Autom. Sin.* **2020**, *8*, 1177–1187. [CrossRef]
40. Qian, G.; Li, Y.; Peng, H.; Mai, J.; Hammoud, H.; Elhoseiny, M.; Ghanem, B. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 23192–23204.
41. Su, H.; Jampani, V.; Sun, D.; Maji, S.; Kalogerakis, E.; Yang, M.H.; Kautz, J. Splatnet: Sparse lattice networks for point cloud processing. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 2530–2539.
42. Xu, Y.; Fan, T.; Xu, M.; Zeng, L.; Qiao, Y. Spidercnn: Deep learning on point sets with parameterized convolutional filters. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 87–102.
43. Yi, L.; Kim, V.G.; Ceylan, D.; Shen, I.C.; Yan, M.; Su, H.; Lu, C.; Huang, Q.; Sheffer, A.; Guibas, L. A scalable active framework for region annotation in 3d shape collections. *ACM Trans. Graph. (Tog)* **2016**, *35*, 1–12. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

sensors

MDPI

Article

Comparison of Point Cloud Registration Techniques on Scanned Physical Objects

Menthy Denayer [1,2,*], Joris De Winter [1,2], Evandro Bernardes [1,2], Bram Vanderborght [1,3] and Tom Verstraten [1,2,*]

[1] Robotics & Multibody Mechanics Group, Vrije Universiteit Brussel, Pleinlaan 9, 1050 Brussels, Belgium; joris.de.winter@vub.be (J.D.W.); evandro.bernardes@vub.be (E.B.); bram.vanderborght@vub.be (B.V.)
[2] Flanders Make, Pleinlaan 9, 1050 Brussels, Belgium
[3] IMEC, Pleinlaan 9, 1050 Brussels, Belgium
* Correspondence: menthy.denayer@vub.be (M.D.); tom.verstraten@vub.be (T.V.)

Abstract: This paper presents a comparative analysis of six prominent registration techniques for solving CAD model alignment problems. Unlike the typical approach of assessing registration algorithms with synthetic datasets, our study utilizes point clouds generated from the Cranfield benchmark. Point clouds are sampled from existing CAD models and 3D scans of physical objects, introducing real-world complexities such as noise and outliers. The acquired point cloud scans, including ground-truth transformations, are made publicly available. This dataset includes several cleaned-up scans of nine 3D-printed objects. Our main contribution lies in assessing the performance of three classical (GO-ICP, RANSAC, FGR) and three learning-based (PointNetLK, RPMNet, ROPNet) methods on real-world scans, using a wide range of metrics. These include recall, accuracy and computation time. Our comparison shows a high accuracy for GO-ICP, as well as PointNetLK, RANSAC and RPMNet combined with ICP refinement. However, apart from GO-ICP, all methods show a significant number of failure cases when applied to scans containing more noise or requiring larger transformations. FGR and RANSAC are among the quickest methods, while GO-ICP takes several seconds to solve. Finally, while learning-based methods demonstrate good performance and low computation times, they have difficulties in training and generalizing. Our results can aid novice researchers in the field in selecting a suitable registration method for their application, based on quantitative metrics. Furthermore, our code can be used by others to evaluate novel methods.

Keywords: point cloud registration; digital twins; CAD model alignment; point cloud datasets

1. Introduction

Point cloud registration (PCR) is used in applications like building information modelling [1], augmented reality authoring [2] and robotics [3]. The problem of PCR consists in finding the (rigid) transformation between two point clouds, the source and the template, minimizing a cost function. These point clouds can be sampled from available CAD models or generated using stereovision [4] and laser-scanning techniques, including LiDAR [5]. In practical applications such as manufacturing, a combination of CAD models and (partial) scanning data is often used.

A closed-form solution exists when correspondences between the two point clouds are exactly known [6]. This is the case, for example, when using synthetic data, generated on a computer. However, these correspondences are unknown and not exact when working with real-world scans or point clouds captured by different sensors. Synthetic datasets approximate this by adding Gaussian noise [7] or removing a part of the object (partiality) [8,9]. Still, these approximations fail to capture real-world 3D-scans, as they lack details like the rounding of sharp corners, deformations, density variations or slight scaling. We believe the community could benefit from a comparison with point clouds sampled from CAD models and generated using laser-scanning. However, such comparisons are lacking in the existing literature.

Review papers on PCR techniques, metrics and datasets are presented in the literature [10–15]. These typically use synthetic datasets [12], such as the Stanford Bunny [16]. Comparisons on real-world scans exist for LiDAR data [5], but existing datasets are mostly limited to indoor and outdoor environments [17,18], instead of specific objects [19–21]. Aside from the MVTEC ITODD dataset [22], which focuses on industrial object scans, other datasets mainly include everyday objects or environments. To address this gap, we collect new real-world data, based on the Cranfield benchmark [23], containing basic geometries. These basic objects facilitate the creation of clear 3D scans while introducing challenges such as symmetric solutions, noise and partiality.

Aligning the captured point clouds with their CAD models is often done using the standard ICP method in the literature. [4,24–26]. However, this typically involves using a (fully) scanned template model [24,26,27], instead of sampling from a CAD model. While popular, ICP's performance is sensitive to the initial pose of both objects and can easily fall into local minima solutions [10]. Thus, there is a need to look for other, more robust registration methods.

Classifications of PCR methods have been proposed in the literature [28]. Ref. [29] compares different feature descriptors for ICP-based methods and RANSAC-based methods (SAC-IA). They find the Fast Point Feature Histogram (FPFH) to be accurate and fast. For this reason, it is also used in this paper. However, we extend the comparison to include other methods, like FGR and deep learning methods. Comparing several techniques is crucial to expand beyond the basic ICP method in practical applications. As indicated by [28,30], there is still a large reliance on classical methods like ICP and NDT, while benchmarks for learning-based methods and pretrained models for real-life scenarios are lacking. The comparison by [31] focuses on RANSAC-based methods and inlier IG-methods. Ref. [32] includes recent deep learning methods such as SpinNet [33] and a graph-based method, TEASER [6]. However, the comparison is again performed on range scans, instead of solving the CAD alignment problem. Finally, ref. [34] compares deep-learning-based registration methods based on previously published metrics. We compare six different registration methods, including RANSAC, FGR, PointNetLK and RPMNet, which are important, well-known global and learning-based registration techniques [35]. Alternative methods exist in the literature, like probabilistic methods (Deep-GMR [36], NDT [37], CPD [38]), graph-based methods (TEASER [6]) and other learning-based methods (DeepPro [39], SpinNet [33], REGTR [40]). These are considered to be out of scope for this paper. We focus instead on classical and learning-based techniques. However, the created code allows others to evaluate their performance using the same methodology on a given dataset.

In this paper, CAD model alignment is used to compare six popular registration methods, including GO-ICP [8], RANSAC [41], FGR [42], PointNetLK [43], RPMNet [9] and ROPNet [44]. The point clouds are generated from an available CAD model and a 3D-scan, created using the Intel RealSense D435i camera. New scans are created based on the Cranfield benchmark dataset. The scans contain noise and outliers, which are typical challenges in PCR, to verify performance in real-world applications. The following assumptions are made:

1. Each point cloud consists of a single object that is already segmented from the environment. However, we adapt the quality of the cutout as a parameter.
2. We already assign each point cloud a label corresponding to the represented object.
3. We do not consider large deformations and shearing [45,46].

The main contributions of this paper are:

1. A comparison of the performance of six registration methods, applied on real-world scans of 3D-printed objects, with relatively basic geometries, and their CAD models.
2. A dataset consisting of a series of real-world scans, based on the Cranfield benchmark dataset [23] with available ground-truth estimation. The Python code, used to run the experiments, and the point cloud scans with their ground truth, are available

at https://github.com/Menthy-Denayer/PCR_CAD_Model_Alignment_Comparison.git (accessed on 14 February 2024).

The remainder of this paper is organized as follows. Section 2, Methodology, details the methodology, including the chosen registration methods, metrics, datasets and ground-truth estimation used to assess the performance. The results of the experiments are presented in Section 3, Results, and discussed in Section 4, Discussion. Finally, Section 5, Conclusion, contains conclusions and future work opportunities.

2. Methodology

2.1. Registration Methods

We selected registration methods based on the following criteria:

- Robustness to noise. Three-dimensional cameras were used to create point clouds. Working in nonoptimal lighting conditions or cluttered environments results in measurement errors and noise. This leads to deformations of the scan compared to the real object, making it more difficult to find correspondences for the registration methods.
- Robustness to partiality. Since we used a single camera in this paper, the object was only visible from one perspective. As a result, the captured point cloud was incomplete, missing the parts of the object, which the camera could not see.
- Limited computation time. For real-time applications, the computation time for the registration process has to be limited. Timing can also be an important aspect to consider when training the learning-based methods.
- Ability to generalize to different objects. The different methods have to work on a wide variety of objects to be widely applicable. Learning-based methods are trained on available CAD models and risk overfitting. Non-learning-based methods can generalize better, which may come at the cost of a lower performance.

These criteria are typically used in the literature to describe the advantages and disadvantages of the different algorithms. We favoured open-source codes to adapt the methods into the comparison framework. Open3D [47] provides an open-source library including the RANSAC, FGR and ICP registration methods. These are standard and popular methods, often used in real-world applications [3,26,48]. Furthermore, we selected learning-based methods for their improved robustness and accuracy when dealing with extensive real-world scans [10,39,49]. Additionally, there is a need to benchmark these methods in the literature [30]. The selected techniques are shortly discussed in Sections 2.1.1 and 2.1.2.

2.1.1. Non-Learning-Based Methods

Non-learning-based methods do not have to be trained and are therefore quick to set up and use. We implemented GO-ICP from the author's code [8], and we used the Open3D implementation for RANSAC and FGR [47].

- GO-ICP [8] improves upon the standard ICP method by finding the global optimum solution. ICP-based methods solve the registration problem by minimizing a cost function. These algorithms typically establish correspondences based on distance. GO-ICP is robust to noise. However, the method tends to be slow.
- RANSAC [41] is a global registration method, often used in scene reconstruction [48]. It uses the RANSAC algorithm to find the best fit between the template and source point clouds. Features are extracted using Fast Point Feature Histogram (FPFH) [50], which is a point-based method [32]. RANSAC is robust to outliers and noise and does not require any training process. However, it requires a preliminary step for feature extraction and there are multiple parameters to tune.
- FGR [42] is a fast registration method, requiring no training. It also uses FPFH to extract features but does not recompute the correspondences during the execution. It can perform partial registration but is more sensitive to noise.

We selected GO-ICP as it provides robust, high-accuracy results. Thus, it is interesting to compare its outcomes to methods like FGR and RANSAC, which are much faster, but less reliable.

2.1.2. Learning-Based Methods

Learning-based methods are trained using a dataset to extract features and compute the transformation matrix. The training process consists of creating many iterations of a template and a transformed source. Each time the registration problem is solved and compared to the ground-truth solution. An error is then computed to adjust the weights of the network. We implemented PointNetLK and RPMNet from available codes [51], while we took ROPNet from the author's code [44].

- PointNetLK [43] is a learning-based method. It uses PointNet to generate descriptors for each point. This information is then used to compute the transformation matrix through training. The method is robust to noise and partial data. However, the performance drops when the method is applied to unseen data and for large transformations.
- RPMNet [9] is another learning-based method. It combines the RPM method with deep learning. RPM itself builds upon ICP, using a permutation matrix to assign correspondences. The transformation matrix is computed using singular value decomposition (SVD). RPMNet is robust to initialization and noise, and also works for larger transformations. RPMNet is, however, reported by [9] to be slower than other methods like ICP or DCP.
- ROPNet [44] is a learning-based method, created to solve the partial registration problem. First, a set of overlapping points is established. Afterwards, wrong correspondences are removed, turning the partial-to-partial into a partial-to-complete problem. Finally, SVD is used to compute the transformation matrix. It is robust to noise and can generalize well.

PointNetLK forms an important milestone for deep-learning-based PCR methods, while RPMNet and ROPNet are promising novel approaches. We selected these methods as each one approaches the problem of PCR differently, resulting in different training capabilities and accuracies.

2.2. Metrics

We used four groups of metrics to compare the registration methods. The first group of metrics expressed the errors in degrees and a unit of length. They were the easiest to interpret and yielded a direct evaluation of the rigid transformation. These metrics included the mean absolute error (MAE) [9], mean relative error (MRE) [13] and root-mean-square error (RMSE) [36]. All metrics could express both translational and rotational errors, in a unit of length and degrees, respectively.

A second group of metrics evaluated the accuracy of the alignment. This included the recall [36] metric and the coefficient of determination R^2 [52]. The metrics were expressed as a number between zero and one or as a percentage. The better the alignment, the higher their value.

In some cases, the registration may lead to unsatisfactory results. In these cases, the absolute values of the errors are not as relevant. However, it is interesting to save the number of failure cases [6,21], which made up the third group of metrics, and the scans for which they occurred. Thus, whenever a result met the condition of $R^2 < 0$ or $MRAE > 120°$, it was considered to be a failure. An example of a negative R^2 result is shown in Figure 1. The condition aims at removing only the extreme cases of misalignment, where the found transformation is small. Thus, in these cases, ICP refinement cannot correct the results.

Finally, we also recorded the registration time as a metric. It was counted whenever the method is called upon, with the source and the template already loaded, until the transformation matrix was returned.

We implemented the evaluation of the results in Python. The code can be found at https://github.com/Menthy-Denayer/PCR_CAD_Model_Alignment_Comparison.git (accessed on 14 February 2024).

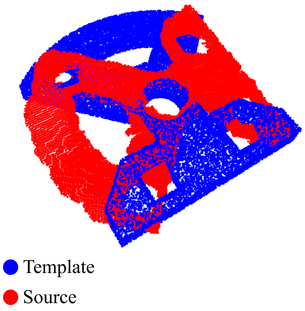


Figure 1. An example of a solution with a negative R^2 value. The estimated transformation (red) does not overlap the template (blue). The result shown is for the base-top plate object, after applying PointNetLK.

2.3. Materials

We used an Intel (USA) RealSense D435i camera to capture the point cloud scans. It can capture depth at 30 cm, yielding high-accuracy scans for the considered objects. The datasheet indicates an absolute error (z-accuracy) of ±2% for objects captured within 2 m from the camera. The spatial noise (RMSE) is less than 2%. The D435i camera has a 1280 × 720 depth resolution. We processed the captured point clouds manually using the RealSense viewer application, separated them from the environment and labelled them. We sampled the template point clouds from the corresponding CAD files with twice the number of points from the source (captured point cloud). We did this to obtain a similar point cloud density, considering partiality. The template was scaled to match the size of the captured point cloud.

For the experiments, we used two datasets (Figure 2): the Cranfield benchmark [23] and the ModelNet40 [53] dataset. The Cranfield benchmark is used to assess, for example, robotic peg-in-hole (PIH) manufacturing operations. It contains six unique objects with basic geometries. The objects all have at least one symmetry axis, as shown in Figure 3. This means multiple ground-truth solutions exist, which were considered in the comparison. We 3D-printed the objects to reduce reflections, as these were not considered in this paper. The largest and smallest objects had a characteristic length of 22 cm and 6 cm, respectively.

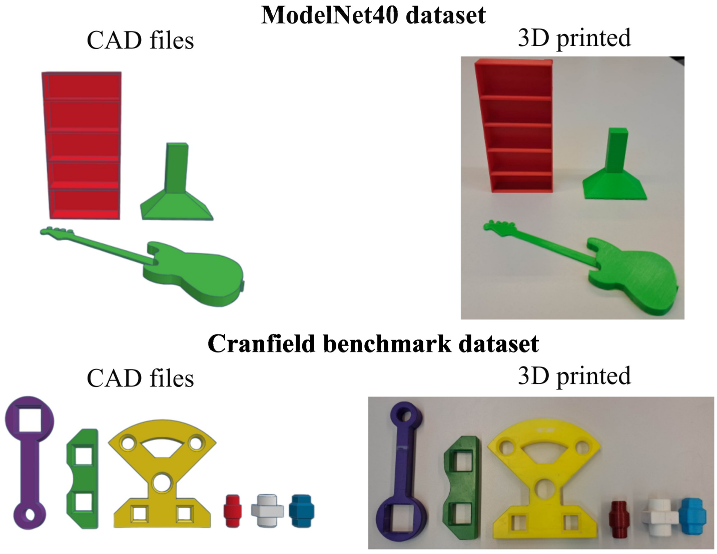


Figure 2. Objects from the ModelNet40 dataset and Cranfield benchmark used during the experiments.

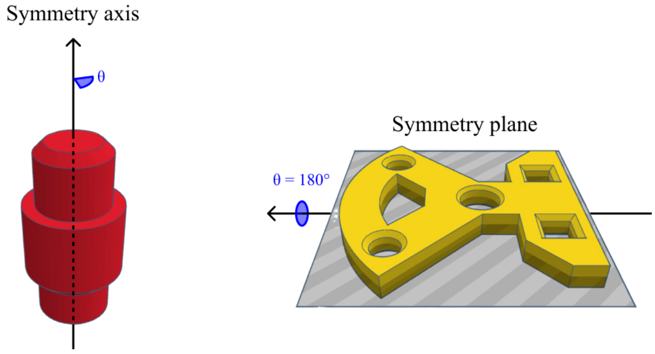


Figure 3. Objects from the Cranfield benchmark dataset contain a symmetry axis or symmetry plane, resulting in an infinite number of ground-truth solutions.

The ModelNet40 dataset is typically used to evaluate point cloud registration methods on synthetic data. The dataset consists of multiple models in 40 categories of objects. We selected and 3D-printed three objects with simple geometries. Training data were also available for the learning-based methods [51].

We placed the objects flat on a light table, as shown in Figure 4, in different orientations. The camera was fixed above the table. Depending on the object, we brought the camera closer or farther to obtain a clear point cloud scan. The objects were angled at 45° or 90°. We only added scans when the general shape of the object was sufficiently recognizable.

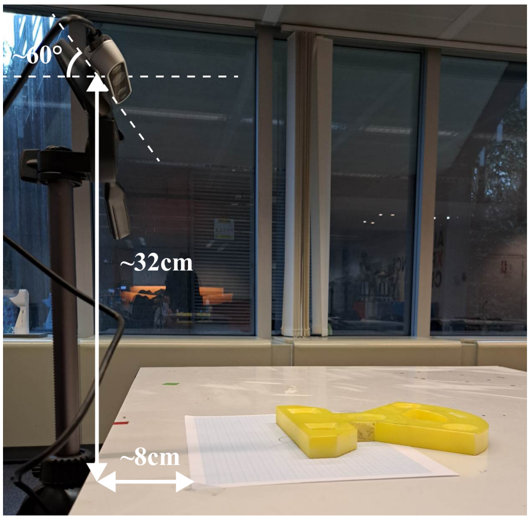


Figure 4. The experimental setup used for capturing the point clouds. The 3D camera is fixed. The paper is used as a reference for orienting the 3D-printed objects.

We also used the datasets for training the learning-based methods, selected in Section 2.1.2. The training data were generated synthetically, as creating a sufficiently large dataset with real scans is very time-intensive. We sampled point clouds from the CAD models and randomly transformed them according to literature guidelines [7,9,43,44,52,54]. Additional variations were introduced in the data by adding noise, partiality, a floor or a combination. Due to convergence issues, not all methods could be trained on all datasets. All methods were trained on the normal and noisy ($\sigma = 0.01$) datasets. RPMNet and ROPNet were trained on partial data, where 50% and 70% of the points were retained, respectively. The same methods were trained on the floor, noisy and partial dataset, with a noise level of 0.01 and 70% of points retained.

2.4. Ground-Truth Estimation

To compute the metrics from Section 2.2, we needed to estimate the ground truth. The process was based on using known information, like the object's orientation on the table, estimated values such as the normal vector on the table and finally, a visual correction for the translation [6]. We placed the 3D camera parallel to the table, which meant the x-axis of the template was too. Figure 5 shows the steps, which are detailed below.

1. We centred both point clouds on the origin by subtracting their mean.
2. Using an estimation of the normal vector on the table, we performed a rotation around the template x-axis, aligning the y-axes of both objects.
3. Given the known rotation of the object on the table, we performed a final rotation around the new y-axis.
4. We performed visual corrections for the rotation and mainly translation, similar to [6].

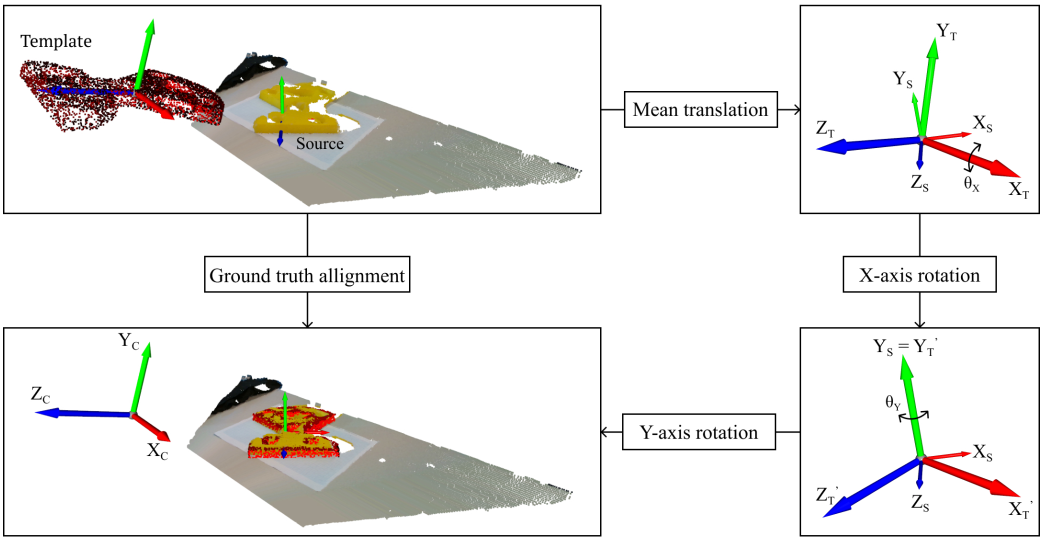


Figure 5. Steps performed to estimate the ground-truth estimation. The red and yellow point clouds represent the template and source, respectively. $X_T Y_T Z_T$ is the template's and $X_S Y_S Z_S$ the source's coordinate system.

Appendix A (Table A1) details the validation of the ground truth, showing an accuracy of around 2 mm and 3°. The processed point clouds and their ground-truth transformations are available at https://github.com/Menthy-Denayer/PCR_CAD_Model_Alignment_Comparison.git (accessed on 14 February 2024).

2.5. Registration Parameters

Each registration method has a range of parameters that can be tweaked. The considered parameters are given in Table 1 for each method.

The zero-mean parameter refers to the centring of the point clouds, which reduces the transformation size. This parameter was considered as it is a simple pre-processing step, removing the offset from the camera.

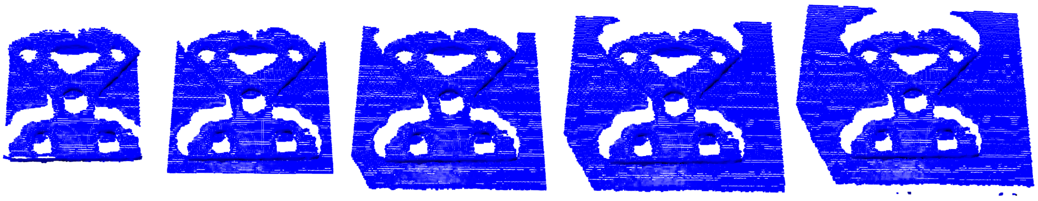
For GO-ICP, the MSE threshold and trim fraction need to be defined. The MSE threshold determines the convergence criteria. The trim fraction determines the fraction of outliers to be removed.

The voxel size is a simple filtering technique, used to downsample the point clouds. It takes the average of all points inside a small voxel, with the voxel size indicating its scale. As this parameter reduces the number of points in the point cloud, it is interesting to consider its effect on the computation time.

Table 1. Parameters checked for each method. A check mark (✓) or range indicates the parameter is verified, a dash (/) means the parameter is not applicable and a lightning symbol (↯) indicates the parameter is checked, but no convergence was reached. Training models are reported in Section 2.3.

Method	GO-ICP	RANSAC	FGR	PointNetLK	RPMNet	ROPNet
Zero mean	✓	✓	✓	✓	✓	✓
Refinement	✓	✓	✓	✓	✓	✓
Bounding box	↯	$1 \rightarrow 1.8$				/
Voxel size [m]	$10^{-3}, 10^{-2}$	$10^{-4} \rightarrow 10^{-1}$	$0, 10^{-3} \rightarrow 10^{-2}$			
MSE Threshold [m]	$10^{-5} \rightarrow 10^{-1}$	/	/	/	/	/
Trim fraction [/]	$10^{-4} \rightarrow 10^{-1}$	/	/	/	/	/
Training model	/	/	/	✓	✓	✓

We adapted the bounding box to simulate the effect of different cutouts around the object, as in Figure 6. The larger the bounding box, the more environmental information is included in the point cloud. However, methods that can work for larger bounding boxes require less pre-processing. This is especially interesting in real-time applications. Furthermore, the maximal bounding box places additional requirements on preliminary object detection and filtering steps.



BB=1.0 BB=1.2 BB=1.4 BB=1.6 BB=1.8

Figure 6. Effect of adapting the bounding box to increase the information around the object in the point cloud.

Finally, training models were varied for the learning-based methods. We used the Cranfield benchmark first to train the methods, as mentioned in Section 2.3. Pre-trained ModelNet40 training models were compared to the results achieved using the Cranfield benchmark.

Finally, we used ICP refinement to refine the results, similar to [6]. ICP [43] is often used in applications for its low computation time and simplicity. The main disadvantages of ICP include its lower performance for larger transformations and susceptibility to local minima.

2.6. Data Processing

We applied the registration methods, selected in Section 2.1, on the created point cloud scans, to align them with their templates. For each object, we created multiple scans. The alignment was repeated several times to verify whether the same results were obtained. Furthermore, we varied several parameters for each registration method, as mentioned in Section 2.5. From these experiments, we selected the parameters leading to the best result, over the performed experiments, per object. This means the lowest values for the MAE, MRE, RMSE and number of failure cases and the highest values for the recall and R^2 metric. We then averaged the metrics for a final comparison of the methods. Figure 7 gives a schematic overview.

The standard deviation σ is the square root of the sum of experimental variances, weighted by the number of samples, not considering the failure cases. For the number of failure cases and time, we used the standard formula for the variance, with the sum taken over the different objects instead.

We ran our experiments on an HP Omen (USA) Windows laptop with an NVIDIA GeForce RTX 3070 GPU and AMD Ryzen 7 processor.

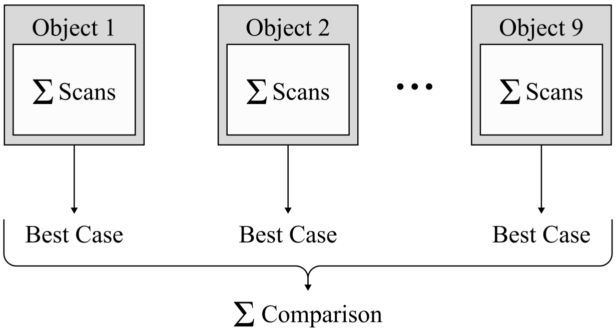


Figure 7. For each object, the results of the different scans were averaged (Σ). The best cases for each object, in terms of the parameters, were selected and averaged, to compare the different methods.

3. Results

3.1. Training Validation Results

We validated the learning process on a test dataset for the Cranfield models. Table 2 gives an overview. Since all points of the point clouds were rescaled to fit into a unit sphere, to comply with training standards [7,9,43,44,52,54], the synthetic objects were on the order of metres, compared to centimetres for the real, 3D-printed objects. Table 2 shows a largest error of 25.76 mm, which corresponds to 1.2% of the largest object's characteristic length. Training validation indicated a good training convergence for all datasets. However, the recall values were lower, specifically for RPMNet and ROPNet, compared to PointNetLK. R^2 values were 1.00, indicating a perfect overlap. However, when working with large numbers, this metric is more prone to numerical rounding errors.

Table 2. Validation results of training PointNetLK, RPMNet and ROPNet on (1): normal data, (2): noisy data, (3): partial data, (4): floor, noisy and partial data. Training–test iterations were 820 and 205, respectively. The data used for training are also mentioned (A: all, L: limited). The recall limit was 0.01 m. The arrows indicate whether the metric should be as small (↓) or high (↑) as possible for a good registration result.

Dataset	Data [A/L]	MRAE [°] ↓	MRTE [mm] ↓	RMSE [°] ↓	RMSE [mm]↓	MAE [°] ↓	MAE [mm] ↓	Recall [%] ↑	R^2 [/] ↑
PointNetLK									
1	A	2.13	0.78	0.07	0.05	0.37	0.00	91.71	1.00
2	L	0.31	2.94	0.00	2.88	0.13	0.01	97.82	1.00
RPMNet									
1	A	2.60	3.00	0.04	2.23	0.82	0.00	70.76	1.00
2	A	2.90	3.16	0.05	2.14	0.67	0.00	72.82	1.00
3	A	2.71	16.42	0.04	10.52	0.85	0.11	49.78	1.00
4	A	2.40	25.76	0.02	17.94	0.98	0.32	30.40	1.00
ROPNet									
1	A	0.02	0.00	0.00	0.00	0.00	0.00	100.00	1.00
2	A	0.24	1.76	0.00	1.22	0.11	0.00	99.91	1.00
3	A	0.99	8.83	0.02	11.95	0.49	0.14	79.67	1.00
4	A	1.12	11.51	0.01	7.33	0.48	0.05	69.58	1.00

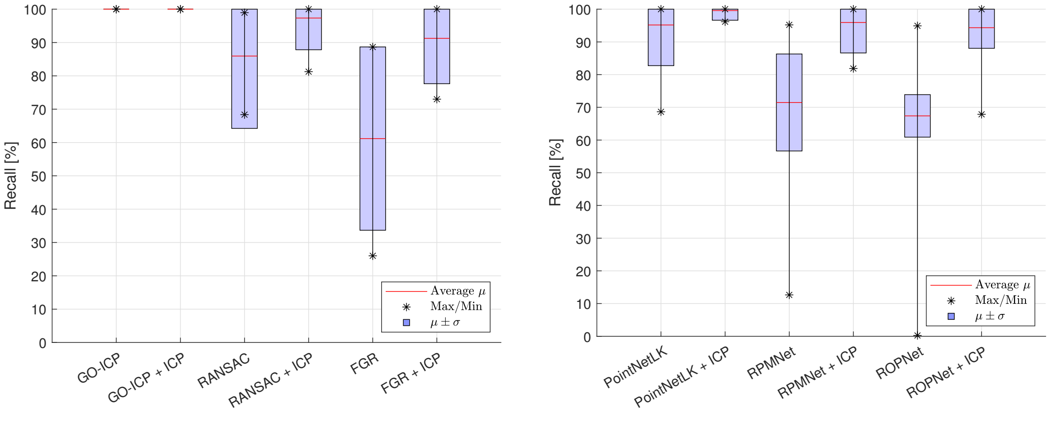
3.2. PCR Methods Comparison

We compared the methods by selecting a representative metric for key criteria, such as precision, variance, speed, generalizability and required pre-processing, as shown in Table 3. An overview of all metrics for each method can be found in Appendix B (Table A2).

The zero-mean method had little effect on the results. The best case was used for each method. GO-ICP, RPMNet and ROPNet were not centred (nonzero mean), while RANSAC and FGR were centred (zero mean). PointNetLK centred the point clouds automatically.

The lower the MSE threshold, the more accurate the results. As the value increased, the number of failure cases rapidly rose until all scans led to unsatisfactory results. However, the registration time increased on average by 10 s when lowering the threshold from 10^{-1} m to 10^{-5} m, with the largest object taking 67 s to solve. The trim fraction had little effect on the results, as the point clouds were already cleaned in pre-processing.

The R^2 value and recall indicate the precision, as shown in Table 3 and Figure 8, respectively. GO-ICP achieved the highest accuracies, where a successful alignment usually coincided with recall values of 100% and $R^2 = 0.98$. RANSAC, PointNetLK and RPMNet, combined with ICP refinement, also resulted in precise alignments, though with increased variance, as seen in Figure 8. FGR was less precise, but one of the faster methods, as shown in Table 3. ROPNet scored well in terms of the recall and R^2 metrics, but led to a high number of failure cases, as shown in Figure 9, even after applying refinement. Figure 8 shows the effect of applying ICP refinement. For all methods, the recall metric improved significantly, while variances dropped. GO-ICP was the only exception, where ICP did not significantly improve the results.



(**a**) (**b**)

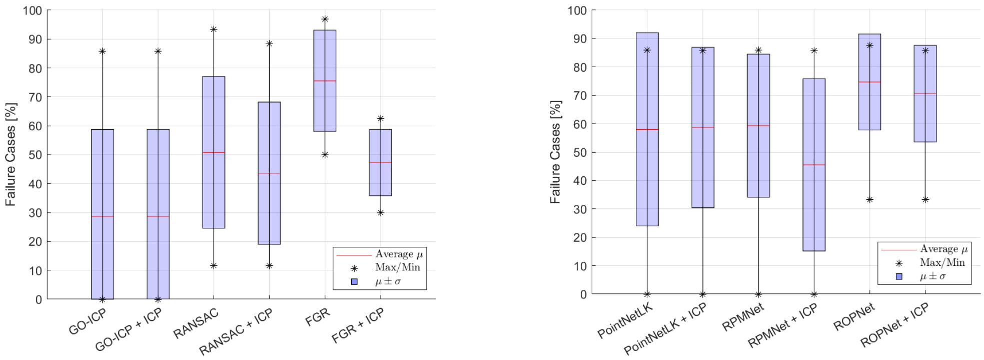
Figure 8. Average recall metric, with average maximal and minimal values indicated over the different objects, as well as the region of variation $\mu \pm \sigma$. (**a**) Non-learning-based methods. (**b**) Learning-based methods.

Computation times were highest for GO-ICP, as shown in Figure 10. The other registration methods took on average less than 1 s to solve. FGR was among the fastest methods on average but showed a larger spread. The learning-based methods were quick to solve the registration problem.

Figure 9 presents the number of failure cases. A large variation is visible for all methods over the different objects and scans. On average, GO-ICP led to the lowest number of failure cases, as also indicated in Table 3. In contrast, ROPNet showed the highest number of failure cases on average. Refinement had a small effect on that metric.

Finally, Table 3 highlights the pre-processing criterion. This refers to the required cleaning of the initial point cloud for the method to work successfully. The recall metric for a bounding box 80% larger than the cleaned-up point cloud was used as a representative metric. GO-ICP failed to converge when too much background clutter was included. PointNetLK and FGR yielded a result, though the registration quality was strongly reduced as the bounding box was increased, with recall values of only 0% and 12.65%, respectively.

RANSAC and RPMNet both led to recall values above 20%. However, the methods performed the best when the point cloud was filtered from outliers and environmental clutter.



(**a**) (**b**)

Figure 9. Average number of failure cases, with average maximal and minimal values indicated over the different objects, with the region of variation $\mu \pm \sigma$. (**a**) Non-learning-based methods. (**b**) Learning-based methods.

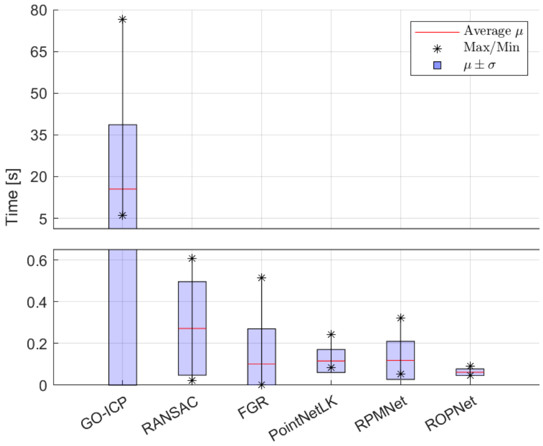


Figure 10. Average computation time for all methods, with average maximal and minimal values indicated over the different objects, with the region of variation $\mu \pm \sigma$. GO-ICP leads to much higher registration times, hence the jump in values.

Table 3. For each criterion, a representative metric was chosen. ICP refinement was not applied during the experiments for pre-processing and speed. However, ICP refinement was included for all other values. Dash (/) indicates missing data. Lightning (↯) indicates no convergence.

Method ↓	**Precision**	**Variance**	**Speed**	**Generalizability**	**Pre-processing**
Metric →	R^2 ↑	$\sigma(R^2)$ ↓	**Time [s] ↓**	**Failure [%] ↓**	**Recall [%] for *BB* = 1.8**
GO-ICP	0.98	0.04	15.50	28.64	↯
RANSAC	0.93	0.16	0.27	43.58	21.70
FGR	0.81	0.20	0.10	47.26	12.65
PointNetLK	0.97	0.15	0.12	58.65	0
RPMNet	0.89	0.22	0.12	45.52	48.87
ROPNet	0.95	0.12	0.06	70.60	/

4. Discussion

4.1. Registration Parameters

The voxel size parameter significantly affected the registration results for RANSAC, FGR, ROPNet and RPMNet. For these methods, the recall metric dropped from >80% to <20%, for a change in the voxel size of only 2 mm. The parameter also did not show a clear trend. The optimal voxel size changes depended on the object, method and metric. PointNetLK was more indifferent to the voxel size, as the results remained consistent for almost all objects. The best results were found for a voxel size of around 1 cm. Finally, we combined GO-ICP with the voxel size. However, it is important to adapt the MSE threshold parameter accordingly, otherwise, the method no longer converges. Adapting the voxel size can improve the results for a higher MSE threshold. Most objects, leading originally to a zero recall, resulted in a recall of >60% after adapting the voxel size.

We found that the registration accuracy typically decreased with an increasing bounding box. Some methods, such as FGR and PointNetLK, were more sensitive, with the recall metric decreasing from 50% to 10% and 90% to 0% as the bounding box was increased by 80%. Meanwhile, RPMNet and RANSAC converged more slowly to a lower recall value. GO-ICP, on the other hand, no longer converged when the bounding box was increased. As a general guideline, the best results were obtained when the object's information was maximized and the environmental clutter was minimized. Thus, pre-processing is important to a successful alignment. Furthermore, this limits the maximal size of the bounding box extracted from preliminary object detection steps.

For the training validation, the lower recall values can be explained by the scaling. The limit for recall was chosen at 1 cm, while the average translational error was larger in these cases. Additionally, the R^2 metric is more sensitive to numerical rounding errors, especially when working with larger values. Still, all other metrics showed the good performance of these methods on the synthetic datasets. On the point cloud scans, the normal datasets led to the best results in terms of accuracy. The Cranfield benchmark could generalize well, even to the considered ModelNet40 objects. Thus, adding Gaussian noise or partiality did not directly lead to improved results. This indicates the need for a better model of the captured point cloud scans to improve the training dataset.

For all methods, ICP refinement led to significant improvements over all objects, with rotational and translational errors decreasing more than 10° and 5 mm, respectively. The only exception to this trend was GO-ICP, where the relative gain of applying refinement diminished from 90% and 80% to almost 0% for the MRAE and MRTE, respectively, when decreasing the MSE threshold. The highest recorded registration time for ICP was 0.32 s, for the largest object, but it typically ranged on the order of 10^{-2} s for most other cases.

4.2. PCR Methods Comparison

Of all methods, GO-ICP achieved the best performance in terms of precision. The major downside of GO-ICP lay in the higher registration time of multiple seconds up to a minute, making it insufficient for real-time applications. This agreed with the literature [8], where errors of around 5° were obtained on the Stanford dataset. RMSEs were reported of at most 0.05 when applied on synthetic data, while we found errors of 0.04° and 1.58 mm. ICP refinement only took 0.01 s on average, as there was only a small gain.

RANSAC, refined with ICP, reached similar performance to GO-ICP, while showing higher variances. This can be explained by the fact that RANSAC uses a random initialization which can cause different results for the same scan. However, RANSAC was much faster than GO-ICP, with average registration times below 1 s. Furthermore, parameters needed to be correctly set to achieve a high performance as indicated in the literature [41].

FGR was one of the quickest methods and had a similar performance to RANSAC's, although it was less accurate and with more failure cases. The method is also more prone to noise, as indicated by [42]. RMSEs of only 0.008 were achieved on synthetic data, even when adding noise with $\sigma = 0.005$. On a scene benchmark, a recall of only 51.1% was obtained, which was lower than 61.18% found in this paper. We found speeds for FGR around 0.1 s, which was close to the reported 0.2 s in the literature [42].

PointNetLK achieved the best performances overall for the learning-based methods. Like RANSAC and FGR, the results were the best when PointNetLK was refined using ICP. However, the number of failure cases and variability between the results were both higher. PointNetLK yielded the same result for a single scan, leading to slightly lower deviations than RANSAC and FGR. Registration times were limited to 0.12 s and GPU requirements were the lowest among the tested learning-based methods. Still, PointNetLK required an extensive training process with limited ability to generalize to unseen data. Furthermore, the performance dropped when transformations were larger, as also indicated in the literature [43].

For RPMNet, the number of failure cases was lower than for PointNetLK and FGR, showing the higher robustness to initialization, as mentioned in the literature [9]. RPMNet converged more easily on complex datasets compared to PointNetLK. However, GPU requirements were also higher. RPMNet achieved errors smaller than a unit degree or millimetre when applied to clean data. Errors increased slightly when Gaussian noise was added, but were still much below 25.78° and 10.21 mm found on 3D scans.

ROPNet was outperformed by the other methods, with an MRAE of 7.90° and MRTE of 4.89 mm and around 70% failure cases, after refinement. The performance was lower than reported in the literature [44]. On synthetic data, ROPNet could achieve errors of around 1°, even on unseen ModelNet40 data with added noise. This can be related to the datasets not representing the real-world scans sufficiently well. As a result, ROPNet had difficulty generalizing to the scanned data.

Our results show that learning-based methods (PointNetLK, RPMNet) can match the performance of classical methods like RANSAC and FGR. However, the studied techniques also highlight the need to consider trade-offs in accuracy, speed and pre-processing. Practical considerations are further discussed in Section 4.3.

As a final observation, failures typically occurred due to a low point cloud quality, a too large initial transformation or due to the choice of parameters. As a result, we observed high variances over different objects. ICP could refine the successful registration results, but could not find large transformations that would turn a failure into a success. The lower the number of failure cases, the better the method can generalize to different objects and scans.

4.3. PCR Methods Guidelines

An overview of the compared methods is shown in Figure 11, which can serve as a simple guide to aid in selecting the correct registration method for the reader's application.

RANSAC, PointNetLK and RPMNet combined with ICP refinement yield accurate results and fast registration times. However, errors are larger when applying the methods on real-world scans compared to applying them on synthetic data. These methods can be used in applications requiring a real-time estimation of the transformation matrix, while still achieving precise results on most scans. RANSAC and FGR, in particular, require no training and have low computation times and requirements. These methods are most interesting in real-time registration and could benefit from the knowledge of previously found transformations as an initial guess. RANSAC generalizes well to different objects and scans while being less sensitive to environmental clutter.

ROPNet and PointNetLK achieve high speeds after training. Furthermore, RPMNet can generalize to different objects and yields the best results among the tested methods when clutter is included in the point cloud scan. Learning-based methods need training, which requires more time and higher GPU requirements. Moreover, convergence during training is not guaranteed. Further investigation into their training process and ability to generalize are required to achieve high-fidelity results.

GO-ICP achieved the highest accuracies among the evaluated methods, even reaching similar performance to that found in the literature. However, the method was also the slowest, with registration times of multiple seconds. Furthermore, it required a cleaned-up point cloud with limited outliers and environmental clutter. GO-ICP should be considered for applications requiring high-accuracy results, where computation times do not play a significant role, like 3D modelling.

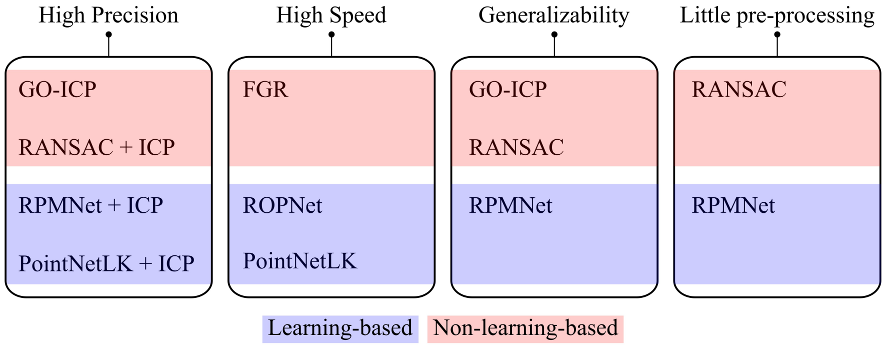


Figure 11. Overview of the compared methods, grouped according to their main strengths.

4.4. Limitations

The present study included only a limited number of registration methods. Future work could focus on extending the comparison, to include state-of-the-art methods such as DeepPro [39], REGTR [40] or TEASER++ [6]. Additionally, classical probabilistic methods, like NDT or CPD, can also be evaluated.

The ground-truth estimation only allowed us to estimate the registration accuracy up to a couple of millimetres and degrees, which might be insufficient for more sensitive applications. Furthermore, differences between methods of less than 2 mm or 3° cannot be considered significant.

The quality of the captured point cloud can be improved by using multiple cameras or a moving one, reducing the partiality in the data. Additionally, filtering [55–58] or point cloud completion [59] techniques can also be added.

Finally, the learning-based methods were trained on a series of basic datasets. Further investigation into training hyperparameters and more complex training datasets is required to achieve higher performance from the learning-based methods. Thus, these methods might achieve higher accuracies, after a more involved training process.

5. Conclusions

Practical applications of point cloud registration still largely rely on classical methods, like ICP. To accelerate the deployment of advanced methods, quantitative validations are essential. This study performed an in-depth comparison of six registration methods, focusing on classical techniques and deep-learning-based solutions.

Furthermore, literature reviews on point cloud registration algorithms are typically performed on synthetic datasets, instead of 3D scans. This paper compared six registration methods, including GO-ICP, RANSAC, FGR, PoinNetLK, RPMNet and ROPNet. Registration was performed on two point clouds, one sampled from a 3D CAD model and the other captured using the Intel RealSense D435i camera.

GO-ICP, as well as RANSAC, RPMNet and PointNetLK combined with ICP, achieved high-precision alignments, with small rotational and translational errors. Refinement had little effect on GO-ICP, hence it was not required. FGR, ROPNet and PointNetLK ranked among the fastest methods tested, each leading to registration times far below 1 s. GO-ICP, RANSAC and RPMNet led to the fewest failure cases, indicating their ability to work robustly for a wide array of scans and objects. Finally, RANSAC and RPMNet showed the most robustness to environmental clutter, thus requiring less pre-processing for the input point clouds.

Our results can be used by novice researchers in the field to select a PCR method for their application, based on quantitative metrics. Furthermore, the dataset and code used during the experiments have been made available to encourage new validation studies and comparisons.

Future work could focus on improving the capture of the point clouds. Here, a single camera was used at a fixed perspective. Instead, multiple cameras or a moving one can capture a more complete point cloud. Advanced filtering techniques can be considered to improve the point cloud quality. Finally, other methods such as DeepPro and Teaser++ show promising results and can be further tested using real-world data.

Author Contributions: Conceptualization, M.D., J.D.W. and B.V.; data curation, M.D.; formal analysis, M.D.; funding acquisition, T.V.; investigation, M.D.; methodology, M.D.; project administration, M.D. and J.D.W.; resources, J.D.W.; software, M.D.; supervision, J.D.W. and B.V.; validation, M.D.; visualization, M.D.; writing—original draft, M.D.; writing—review and editing, M.D., J.D.W., E.B., B.V. and T.V. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially funded by the Flanders Make SBO project "XRtwin-Cost-efficient authoring and adaptive network streaming of interactive XR representations of (large) industrial objects and environments" (SOCMAAK31).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The Python code, used to run the experiments, as well as the point clouds scans with their ground truth, is available at https://github.com/Menthy-Denayer/PCR_CAD_Model_Alignment_Comparison.git (accessed on 14 February 2024).

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results. The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Appendix A. Ground-Truth Validation

We verified the ground truth by applying ICP refinement to the found transformation, similarly to [20]. Table A1 gives an overview of the results, where a voxel size of 1 cm was used to run the ICP algorithm. The metrics showed a small correction made by ICP, indicating a good ground-truth definition. Hence, we used the results without refinement for the experiments.

Table A1. Mean errors (ϵ) and standard deviations (σ), when applying ICP to refine the found ground-truth estimation, for different bounding boxes (BB). A voxel size of 1 cm was used for ICP.

Type	Data	MRAE	MRTE	RMSE	RMSE	MAE	MAE	Recall	R^2
	[ϵ/σ]	[°]↓	[mm] ↓	[°] ↓	[mm] ↓	[°] ↓	[mm] ↓	[%] ↑	[/] ↑
Cleaned point clouds	ϵ	1.85	1.89	0.02	1.09	2.69	0.00	100	1.00
	σ	1.22	1.11	0.01	0.64	7.93	0.00	0.00	0.01
$BB = 1.0$	ϵ	1.50	2.53	0.01	1.46	0.70	0.00	100	1.00
	σ	0.88	1.66	0.01	0.96	0.83	0.01	0.00	0.00
$BB = 1.2$	ϵ	1.24	2.93	0.01	1.69	0.64	0.00	100	1.00
	σ	0.66	1.48	0.01	0.85	0.86	0.00	0.00	0.01
$BB = 1.4$	ϵ	1.50	2.97	0.01	1.71	0.71	0.00	100	1.00
	σ	0.87	1.62	0.01	0.94	1.07	0.00	0.00	0.00
$BB = 1.6$	ϵ	1.58	3.57	0.01	2.06	0.73	0.01	100	1.00
	σ	0.96	2.34	0.01	1.35	1.07	0.01	0.00	0.00
$BB = 1.8$	ϵ	1.58	3.36	0.01	1.94	0.75	0.01	100	1.00
	σ	0.93	2.17	0.01	1.25	1.08	0.01	0.00	0.00

Appendix B. List of Averaged Metrics

Table A2. Metrics (ϵ) and standard deviations (σ) for the studied registration methods. Recall was taken with a 0.01 m threshold. The best metrics are shown in **bold**. The time mentioned for the refinement cases only includes ICP.

Method	Data	MRAE	MRTE	RMSE	RMSE	MAE	MAE	Recall	R^2	Failure	Time
	[ϵ/σ]	[°]↓	[mm] ↓	[°] ↓	[mm] ↓	[°] ↓	[mm] ↓	[%] ↑	[/] ↑	[%] ↓	[s] ↓
Non-learning-based methods											
GO-ICP	ϵ	**4.85**	2.73	**0.04**	1.58	4.00	**0.00**	**100**	**0.98**	**28.64**	15.50
	σ	3.08	1.61	0.02	0.93	4.13	0.01	0.00	0.04	30.08	23.14
GO-ICP + ICP	ϵ	4.85	**2.73**	0.04	**1.58**	**3.99**	0.00	100	0.98	28.64	**0.01**
	σ	3.10	1.58	0.03	0.91	4.14	0.01	0.00	0.04	30.08	0.00
RANSAC	ϵ	18.62	6.23	0.16	3.60	16.43	0.02	85.94	0.88	50.82	0.27
	σ	12.64	4.20	0.09	2.42	31.31	0.03	21.71	0.13	26.24	0.22
RANSAC + ICP	ϵ	5.85	2.83	0.05	1.63	9.70	0.01	97.34	0.93	43.58	0.04
	σ	7.20	2.51	0.06	1.45	29.05	0.02	9.52	0.16	24.63	0.02
FGR	ϵ	27.41	11.67	0.29	6.74	19.08	0.06	61.18	0.58	75.57	**0.10**
	σ	38.52	4.56	0.13	2.63	14.14	0.06	27.50	0.20	17.52	0.17
FGR + ICP	ϵ	12.59	4.95	0.10	2.86	7.81	0.01	91.26	0.81	47.26	0.04
	σ	12.26	3.40	0.10	1.96	7.00	0.02	13.62	0.20	11.46	0.03
Learning-based registration											
PointNetLK	ϵ	14.04	6.36	0.11	3.67	12.55	0.02	95.18	0.86	57.92	0.12
	σ	13.86	1.09	0.11	0.63	34.82	0.01	12.44	0.16	34.05	0.06
PointNetLK + ICP	ϵ	**4.12**	**2.36**	**0.03**	**1.36**	7.80	**0.00**	**99.57**	**0.97**	58.65	0.05
	σ	8.55	1.31	0.07	0.76	36.18	0.00	2.95	0.15	28.24	0.03
RPMNet	ϵ	25.78	10.21	0.21	5.90	11.08	0.04	71.48	0.62	59.31	0.12
	σ	7.77	1.09	0.06	0.63	2.75	0.01	14.83	0.14	25.20	0.09
RPMNet + ICP	ϵ	7.59	3.65	0.06	2.11	**4.52**	0.01	95.94	0.89	**45.52**	**0.04**
	σ	8.05	2.80	0.06	1.62	6.61	0.01	9.31	0.22	30.34	0.03
ROPNet	ϵ	25.88	10.99	0.21	6.35	30.84	0.05	67.38	0.72	74.70	**0.06**
	σ	10.22	2.19	0.08	1.27	9.32	0.02	6.50	0.08	16.90	0.02
ROPNet + ICP	ϵ	7.90	4.89	0.06	2.82	31.91	0.01	94.33	0.95	70.60	**0.04**
	σ	12.31	3.30	0.09	1.91	8.76	0.02	6.29	0.12	17.01	0.05

References

1. Alizadehsalehi, S. BIM/Digital Twin-Based Construction Progress Monitoring through Reality Capture to Extended Reality (DRX). Ph.D. Thesis, Eastern Mediterranean University, İsmet İnönü Bulvarı, Gazimağusa, 2020.
2. Bhattacharya, B.; Winer, E.H. Augmented reality via expert demonstration authoring (AREDA). *Comput. Ind.* **2019**, *105*, 61–79. [CrossRef]
3. Jerbić, B.; Šuligoj, F.; Švaco, M.; Šekoranja, B. Robot Assisted 3D Point Cloud Object Registration. *Procedia Eng.* **2015**, *100*, 847–852. [CrossRef]
4. Ciocarlie, M.; Hsiao, K.; Jones, E.G.; Chitta, S.; Rusu, R.B.; Şucan, I.A. Towards Reliable Grasping and Manipulation in Household Environments. In *Experimental Robotics*; Khatib, O., Kumar, V., Sukhatme, G., Eds.; Series Title: Springer Tracts in Advanced Robotics; Springer: Berlin/Heidelberg, Germany, 2014; Volume 79, pp. 241–252. [CrossRef]
5. Cheng, L.; Chen, S.; Liu, X.; Xu, H.; Wu, Y.; Li, M.; Chen, Y. Registration of Laser Scanning Point Clouds: A Review. *Sensors* **2018**, *18*, 1641. [CrossRef]
6. Yang, H.; Shi, J.; Carlone, L. TEASER: Fast and Certifiable Point Cloud Registration. *IEEE Trans. Robot.* **2021**, *37*, 314–333. [CrossRef]
7. Sarode, V.; Li, X.; Goforth, H.; Aoki, Y.; Srivatsan, R.A.; Lucey, S.; Choset, H. PCRNet: Point Cloud Registration Network using PointNet Encoding. *arXiv* **2019**, arXiv:1908.07906.
8. Yang, J.; Li, H.; Campbell, D.; Jia, Y. Go-ICP: A Globally Optimal Solution to 3D ICP Point-Set Registration. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *38*, 2241–2254. [CrossRef]
9. Yew, Z.J.; Lee, G.H. RPM-Net: Robust Point Matching Using Learned Features. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 11821–11830. [CrossRef]
10. Li, L.; Wang, R.; Zhang, X. A Tutorial Review on Point Cloud Registrations: Principle, Classification, Comparison, and Technology Challenges. *Math. Probl. Eng.* **2021**, *2021*, 1–32. [CrossRef]

11. Stilla, U.; Xu, Y. Change detection of urban objects using 3D point clouds: A review. *ISPRS J. Photogramm. Remote Sens.* **2023**, *197*, 228–255. [CrossRef]
12. Gu, X.; Wang, X.; Guo, Y. A Review of Research on Point Cloud Registration Methods. *Mater. Sci. Eng.* **2019**, *782*, 022070. [CrossRef]
13. Zhang, Z.; Dai, Y.; Sun, J. Deep learning based point cloud registration: an overview. *Virtual Real. Intell. Hardw.* **2020**, *2*, 222–246. [CrossRef]
14. Huang, X.; Mei, G.; Zhang, J.; Abbas, R. A comprehensive survey on point cloud registration. *arXiv* **2021**, arXiv:2103.02690.
15. Huang, X.; Mei, G.; Zhang, J. Cross-source point cloud registration: Challenges, progress and prospects. *Neurocomputing* **2023**, *548*, 126383. [CrossRef]
16. The Stanford 3D Scanning Repository. Available online: https://graphics.stanford.edu/data/3Dscanrep (accessed on 14 September 2023).
17. Zeng, A.; Song, S.; Niessner, M.; Fisher, M.; Xiao, J.; Funkhouser, T. 3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 199–208. [CrossRef]
18. Geiger, A.; Lenz, P.; Urtasun, R. Are we ready for autonomous driving? The KITTI vision benchmark suite. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 3354–3361. [CrossRef]
19. Monji-Azad, S.; Hesser, J.; Löw, N. A review of non-rigid transformations and learning-based 3D point cloud registration methods. *ISPRS J. Photogramm. Remote Sens.* **2023**, *196*, 58–72. [CrossRef]
20. Fontana, S.; Cattaneo, D.; Ballardini, A.L.; Vaghi, M.; Sorrenti, D.G. A benchmark for point clouds registration algorithms. *Robot. Auton. Syst.* **2021**, *140*, 103734. [CrossRef]
21. Osipov, A.; Ostanin, M.; Klimchik, A. Comparison of Point Cloud Registration Algorithms for Mixed-Reality Cross-Device Global Localization. *Information* **2023**, *14*, 149. [CrossRef]
22. Drost, B.; Ulrich, M.; Bergmann, P.; Hartinger, P.; Steger, C. Introducing MVTec ITODD—A Dataset for 3D Object Recognition in Industry. In Proceedings of the 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), Venice, Italy, 22–29 October 2017; pp. 2200–2208. [CrossRef]
23. Abu-Dakka, F.J.; Nemec, B.; Kramberger, A.; Buch, A.G.; Krüger, N.; Ude, A. Solving peg-in-hole tasks by human demonstration and exception strategies. *INdustrial Robot. Int. J.* **2014**, *41*, 575–584. [CrossRef]
24. Hattab, A.; Taubin, G. 3D Modeling by Scanning Physical Modifications. In Proceedings of the 2015 28th SIBGRAPI Conference on Graphics, Patterns and Images, Salvador, Bahia, Brazil, 26–29 August 2015; pp. 25–32. [CrossRef]
25. Decker, N.; Wang, Y.; Huang, Q. Efficiently registering scan point clouds of 3D printed parts for shape accuracy assessment and modeling. *J. Manuf. Syst.* **2020**, *56*, 587–597. [CrossRef]
26. Kumar, G.A.; Patil, A.K.; Chai, Y.H. Alignment of 3D point cloud, CAD model, real-time camera view and partial point cloud for pipeline retrofitting application. In Proceedings of the 2018 International Conference on Electronics, Information, and Communication (ICEIC), Honolulu, HI, USA, 24–27 January 2018; pp. 1–4. [CrossRef]
27. Xu, H.; Chen, G.; Wang, Z.; Sun, L.; Su, F. RGB-D-Based Pose Estimation of Workpieces with Semantic Segmentation and Point Cloud Registration. *Sensors* **2019**, *19*, 1873. [CrossRef]
28. Si, H.; Qiu, J.; Li, Y. A Review of Point Cloud Registration Algorithms for Laser Scanners: Applications in Large-Scale Aircraft Measurement. *Appl. Sci.* **2022**, *12*, 10247. [CrossRef]
29. Liu, L.; Liu, B. Comparison of Several Different Registration Algorithms. *Int. J. Adv. Network, Monit. Control* **2020**, *5*, 22–27. [CrossRef]
30. Brightman, N.; Fan, L. A brief overview of the current state, challenging issues and future directions of point cloud registration. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2022**, *X-3/W1-2022* , 17–23. [CrossRef]
31. Zhao, B.; Chen, X.; Le, X.; Xi, J.; Jia, Z. A Comprehensive Performance Evaluation of 3-D Transformation Estimation Techniques in Point Cloud Registration. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–14. [CrossRef]
32. Xu, N.; Qin, R.; Song, S. Point cloud registration for LiDAR and photogrammetric data: A critical synthesis and performance analysis on classic and deep learning algorithms. *ISPRS Open J. Photogramm. Remote Sens.* **2023**, *8*, 100032. [CrossRef]
33. Ao, S.; Hu, Q.; Yang, B.; Markham, A.; Guo, Y. SpinNet: Learning a General Surface Descriptor for 3D Point Cloud Registration. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 19–25 June 2021; pp. 11748–11757. [CrossRef]
34. Zhao, Y.; Fan, L. Review on Deep Learning Algorithms and Benchmark Datasets for Pairwise Global Point Cloud Registration. *Remote Sens.* **2023**, *15*, 2060. [CrossRef]
35. Qian, J.; Tang, D. RRGA-Net: Robust Point Cloud Registration Based on Graph Convolutional Attention. *Sensors* **2023**, *23*, 9651. [CrossRef] [PubMed]
36. Yuan, W.; Eckart, B.; Kim, K.; Jampani, V.; Fox, D.; Kautz, J. DeepGMR: Learning Latent Gaussian Mixture Models for Registration. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 733–750.

37. Biber, P.; Strasser, W. The normal distributions transform: A new approach to laser scan matching. In Proceedings of the Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453), Las Vegas, NV, USA, 27 October–1 November 2003; Volume 3, pp. 2743–2748. [CrossRef]
38. Myronenko, A.; Song, X. Point-Set Registration: Coherent Point Drift. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 2262–2275. [CrossRef] [PubMed]
39. Lee, D.; Hamsici, O.C.; Feng, S.; Sharma, P.; Gernoth, T. DeepPRO: Deep Partial Point Cloud Registration of Objects. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 11–17 October 2021; pp. 5663–5672. [CrossRef]
40. Yew, Z.J.; Lee, G.H. REGTR: End-to-end Point Cloud Correspondences with Transformers. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 19–24 June 2022; pp. 6667–6676. [CrossRef]
41. Fischler, M.A.; Bolles, R.C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Carthography. *Graph. Image Process.* **1981**, *24*, 381–395. [CrossRef]
42. Zhou, Q.Y.; Park, J.; Koltun, V. Fast Global Registration. In *Computer Vision–ECCV 2016*; Leibe, B., Matas, J., Sebe, N., Welling, M., Eds.; Series Title: Lecture Notes in Computer Science; Springer International Publishing: Cham, Swizerland, 2016; Volune 9906, pp. 766–782. [CrossRef]
43. Aoki, Y.; Goforth, H.; Srivatsan, R.A.; Lucey, S. PointNetLK: Robust & Efficient Point Cloud Registration Using PointNet. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019; pp. 7156–7165. [CrossRef]
44. Zhu, L.; Liu, D.; Lin, C.; Yan, R.; Gómez-Fernández, F.; Yang, N.; Feng, Z. Point Cloud Registration using Representative Overlapping Points. *arXiv* **2021**, arXiv:2107.02583.
45. Ge, X. Non-rigid registration of 3D point clouds under isometric deformation. *ISPRS J. Photogramm. Remote Sens.* **2016**, *121*, 192–202. [CrossRef]
46. Chen, Q.Y.; Feng, D.Z.; Hu, H.S. A robust non-rigid point set registration algorithm using both local and global constraints. *Vis. Comput.* **2023**, *39*, 1217–1234. [CrossRef]
47. Zhou, Q.Y.; Park, J.; Koltun, V. Open3D: A Modern Library for 3D Data Processing. *arXiv* **2018**, arXiv:1801.09847
48. Mahmood, B.; Han, S. 3D Registration of Indoor Point Clouds for Augmented Reality. In *Computing in Civil Engineering*; American Society of Civil Engineers: Reston, VA, USA, 2019; p. 8.
49. Wang, S.; Kang, Z.; Chen, L.; Guo, Y.; Zhao, Y.; Chai, Y. Partial point cloud registration algorithm based on deep learning and non-corresponding point estimation. *Vis. Comput.* **2023**, *Online*. [CrossRef]
50. Rusu, R.B.; Blodow, N.; Beetz, M. Fast Point Feature Histograms (FPFH) for 3D registration. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; pp. 3212–3217. [CrossRef]
51. Sarode, V. Learning3D: A Modern Library for Deep Learning on 3D Point Clouds Data. Available online: https://github.com/vinits5/learning3d (accessed on 4 November 2022).
52. Wang, Y.; Solomon, J.M. PRNet: Self-Supervised Learning for Partial-to-Partial Registration. *arXiv* **2019**, arXiv:1910.12240v2.
53. Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; Xiao, J. 3D ShapeNets: A deep representation for volumetric shapes. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1912–1920. [CrossRef]
54. Wang, Y.; Solomon, J. Deep Closest Point: Learning Representations for Point Cloud Registration. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 3522–3531. [CrossRef]
55. Zhao, J. Point Cloud Denoise, 2023. Original-Date: 2019-05-07T06:25:29Z. Available online: https://github.com/aipiano/guided-filter-point-cloud-denoise (accessed on 30 April 2023).
56. He, K.; Sun, J.; Tang, X. Guided Image Filtering. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 1397–1409. [CrossRef] [PubMed]
57. Han, X.F.; Jin, J.S.; Wang, M.J.; Jiang, W.; Gao, L.; Xiao, L. A review of algorithms for filtering the 3D point cloud. *Signal Process. Image Commun.* **2017**, *57*, 103–112. [CrossRef]
58. Hurtado, J.; Gattass, M.; Raposo, A. 3D point cloud denoising using anisotropic neighborhoods and a novel sharp feature detection algorithm. *Vis. Comput.* **2023**, *39*, 5823–5848. [CrossRef]
59. Wu, H.; Miao, Y.; Fu, R. Point cloud completion using multiscale feature fusion and cross-regional attention. *Image Vis. Comput.* **2021**, *111*, 104193. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

sensors

MDPI

Article

Point Cloud Registration Method Based on Geometric Constraint and Transformation Evaluation

Chuanli Kang [1,2], Chongming Geng [1,*], Zitao Lin [1], Sai Zhang [1], Siyao Zhang [1] and Shiwei Wang [1]

[1] College of Geomatics and Geoinformation, Guilin University of Technology, Guilin 541004, China; 2014012@glut.edu.cn (C.K.); 2120211878@glut.edu.cn (Z.L.); 1020211817@glut.edu.cn (S.Z.); 2120211915@glut.edu.cn (S.Z.); 2120222025@glut.edu.cn (S.W.)

[2] Key Laboratory of Spatial Information and Geomatics, Guilin University of Technology, Guilin 541004, China

[*] Correspondence: 2120211855@glut.edu.cn

Abstract: Existing point-to-point registration methods often suffer from inaccuracies caused by erroneous matches and noisy correspondences, leading to significant decreases in registration accuracy and efficiency. To address these challenges, this paper presents a new coarse registration method based on a geometric constraint and a matrix evaluation. Compared to traditional registration methods that require a minimum of three correspondences to complete the registration, the proposed method only requires two correspondences to generate a transformation matrix. Additionally, by using geometric constraints to select out high-quality correspondences and evaluating the matrix, we greatly increase the likelihood of finding the optimal result. In the proposed method, we first employ a combination of descriptors and keypoint detection techniques to generate initial correspondences. Next, we utilize the nearest neighbor similarity ratio (NNSR) to select high-quality correspondences. Subsequently, we evaluate the quality of these correspondences using rigidity constraints and salient points' distance constraints, favoring higher-scoring correspondences. For each selected correspondence pair, we compute the rotation and translation matrix based on their centroids and local reference frames. With the transformation matrices of the source and target point clouds known, we deduce the transformation matrix of the source point cloud in reverse. To identify the best-transformed point cloud, we propose an evaluation method based on the overlap ratio and inliers points. Through parameter experiments, we investigate the performance of the proposed method under various parameter settings. By conducting comparative experiments, we verified that the proposed method's geometric constraints, evaluation methods, and transformation matrix computation consistently outperformed other methods in terms of root mean square error (RMSE) values. Additionally, we validated that our chosen combination for generating initial correspondences outperforms other descriptor and keypoint detection combinations in terms of the registration result accuracy. Furthermore, we compared our method with several feature-matching registration methods, and the results demonstrate the superior accuracy of our approach. Ultimately, by testing the proposed method on various types of point cloud datasets, we convincingly established its effectiveness. Based on the evaluation and selection of correspondences and the registration result's quality, our proposed method offers a solution with fewer iterations and higher accuracy.

Keywords: geometric constraint; point cloud registration; transformation estimation; evaluation of registration

1. Introduction

Laser sensors utilize the principle of laser ranging to record the three-dimensional coordinates, reflectivity, and texture information from the object being scanned. In practical applications, data are typically acquired from different angles due to limitations imposed by lines of sight, measurement methods, and the geometry of the objects that are being scanned. Each point cloud has its own local reference frames, and the goal of point cloud registration is to unify the multiple angle point cloud data into a common coordinate system [1].

Point cloud registration can be divided into two methods: coarse registration and fine registration [2]. In the context of fine registration methods, the iterative closest point (ICP) algorithm [3] is most commonly used. This algorithm utilizes the least squares method to compute the matching point sets and achieves convergence through iterative optimization, resulting in satisfactory matching results. There were also variants of ICP. The point-to-plane ICP registration algorithm proposed by Low, Kok-Lim et al. [4] demonstrated that using the point-to-plane approach to calculate the transformation matrix is faster and achieves higher registration accuracy compared to the point-to-point approach. S. Rusinkiewicz [5] proposed a symmetric version of the iterative point-to-plane ICP algorithm, which also introduces an alternative method called rotation linearization to simplify the optimization process into a linear least squares problem. ICP is characterized by its straightforward approach and showed its effectiveness, particularly when registering data with a solid initial alignment [6]. Therefore, the challenge of obtaining accurate initial correspondences still persists.

A coarse registration method based on the selection of matching point pairs can be implemented through five steps: keypoint detection, local feature description of keypoints, correspondences selecting, and computation of the optimal transformation matrix [7]. Keypoint detection aims to identify a small set of distinct points from a point cloud to expedite registration, as raw point clouds often contain a large number of points. Local feature description involves using rotation-invariant feature vectors to capture geometric and spatial information within a local surface. By comparing these local geometric features using distance metrics, point-to-point correspondences can be established. However, there are still significant outliers in the initial correspondences. This is due to issues such as keypoint localization, the challenge of distinguishing repeatable patterns using local geometric features, and limited overlap between the point cloud views being aligned. Some of these methods are proposed in order to select out high-quality correspondences [8]. For example, Mian et al. proposed a similarity score based on point cloud descriptors [9]. This method determines correspondences based on the differences between feature elements in point cloud descriptors. However, issues such as data noise, occluded regions, and repeated features, can lead to misjudgment, so this method can only serve as a baseline for evaluation. Another approach is to select high-quality correspondences based on geometric constraints between correspondences. For instance, a base line algorithm is Lowe's ratio rule [10], which determines whether to accept a pair of matching points based on the ratio of the nearest distance to the second-nearest distance. A significant distance difference indicates that the point has better discriminability in the feature space. H. Chen et al. [11] proposed geometric consistency, which imposes constraints based on the geometric distance differences between the source and target points of the two matching points. This approach aims to select correspondences that meet a threshold. Abdullah Lakhan et al. [12] proposed a DAPWTS algorithm framework, which utilizes a secure minimum cut algorithm to partition applications between local nodes and edge nodes. After the application partitioning, an optimal search is performed using a node search algorithm, which also optimizes the structure of point cloud data. However, these methods mostly only utilize a single constraint, leaving room for improvement in the quality of matching point pairs. Moreover, there is the potential of finding the optimal result based on these constraints.

In the field of 3D point cloud registration, there are various algorithms available to solve the outliers' issue. At present, many coarse registration methods were proposed. For example, the compatibility-guided sampling consensus (GC-SAC) registration method proposed by S. Quan and J. Yang [13] utilizes rigidity constraints and salient points' distance constraints for correspondences selection, and determines the optimal transformation matrix based on the maximum number of inliers. Guo et al. [14] proposed a method for point cloud registration using rotation projection statistics (RoPS) features for correspondences feature matching, followed by transformation estimation methods and the ICP algorithm. However, this method relies solely on the initial correspondences generated from descriptors and does not eliminate low-quality correspondences. Another method, proposed by Yang et al. [15], is the consistency voting method, which ranks the correspondences based on constraints such as rigidity and local reference frames. This approach provides a means to prioritize and assess correspondences for improved accuracy and efficiency. Buch et al. [8] proposed a method for confirming inlier points by using Lowe's ratio and the minimum ratio of geometric distances between target points and source points. Sun et al. [16] presented a solution called inlier searching using compatible structures (ICOS), which constructs a compatible structure to facilitate subsequent outlier removal and inlier searching. They designed three efficient frameworks for estimating rotation matrices, known-scale registration, and unknown-scale registration. Rodolà et al. [17] proposed a sparse point matching approach based on game theory; Tombari et al. [18] introduced a method called 3D Hough voting (3DHV), which involves voting based on the positional information of correspondences in the Hough space; Sahloul et al. [19] presented a two-stage voting scheme that uses dense evaluation and ranking of local and global geometric consistency to distinguish inliers. Quan, S. et al. [20] proposed a robust method, progressive consistency voting (PCV), for feature matching in 3D point clouds. It assigns confidence scores to correspondences based on geometric consistency and utilizes a voting-based scheme. Xu, G. et al. [21] proposed a method that combines RANSAC, intrinsic shape signatures (ISS), and 3D shape context descriptor (3DSC) to improve the ICP registration of large point clouds. It uses voxel grid filter for down-sampling, extracts keypoints with ISS, describes them with 3DSC, performs coarse registration with RANSAC using ISS-3DSC features, and achieves accurate registration with ICP. In the subsequent experiments, we will refer to this method as IRIS (improved registration using ISS, RANSAC, and ICP with 3DSC). Yan, L. et al. [22] proposes a graph reliability outlier removal (GROR) method, which is a strategy based on the reliability of the correspondence graph to address the issue of outliers in point cloud registration.

Some methods utilize other characteristics of point clouds for registration, Liang, L. et al. [23] introduce an innovative affine iterative closest point algorithm incorporating color information and correntropy. By integrating color features into traditional affine algorithms, the method established more accurate and reliable correspondences. Liu, J et al. [24] proposed point cloud registration with multilayer perceptrons (PCRMLP), a novel model for urban scene point cloud registration that achieves comparable registration performance to prior learning-based methods. PCRMLP estimates transformations implicitly from concrete instances using semantic segmentation and density-based spatial clustering of applications with noise (DBSCAN) to generate instance descriptors, enabling robust feature extraction, dynamic object filtering, and logical transformation estimation.

In the context of computing the optimal transformation matrix, a commonly used method is the random sample consensus (RANSAC) algorithm proposed by Fischler et al. [25]. This algorithm randomly selects three correspondences from the point sets, generates a rotation and translation matrix, and then counts the inlier points that are within a distance threshold under this transformation matrix. This step is repeated multiple times, and the transformation matrix with the maximum number of inlier points is selected as the final result, while outlier points are discarded. However, the RANSAC method still necessitates a minimum of three correspondences for generating a transformation matrix, and the randomness in selecting these correspondences introduces inaccuracy to its registration result. In contrast, our proposed registration algorithm based on two-point correspondences enhances the precision of the final registration result since these correspondences are filtered through geometric constraints. This approach also saves time compared to the RANSAC, which involves traversing to generate various transformation matrices. Additionally, the sampling process only selects transformation matrix results based on a single criterion, such as a distance threshold, which can potentially miss the optimal transformation matrix. Quan et al. [26] extended the RANSAC algorithm by establishing local reference frames on key points, enabling the inference of transformation matrices with only one correspondence. They also proposed a "maximum overlapping point set" criterion to evaluate the registration results. Another improved variant of the RANSAC algorithm is the optimized sample consensus (OASC) algorithm proposed by Yang et al. [27], which introduces a new error metric. However, these methods only employ a single evaluation criterion for assessing the registration results. One of the research directions of this paper is to explore whether combining multiple evaluation factors would yield improved registration results. By considering various evaluation factors simultaneously, such as alignment accuracy, robustness to outliers, and computational efficiency, it is possible to gain a more comprehensive understanding of the registration performance.

To address the aforementioned issues, this paper proposes a method based on the rigidity and salient points' distance constraints to select out two-by-two correspondences that fit the standard. Then, based on the centroid of two points, we can calculate their translation matrix. Based on the vector of the two points and their salient points, we can construct their two-point rotation pose (reference frame). By combining the two matrices mentioned earlier, we obtain a transformation matrix. With the transformation matrix of the source and target point clouds known, we obtain the transformation matrix of the source point cloud by reverse deduction. Finally, we proposed a transformation evaluation method combined with overlap ratio and inlier points to select the best transformation matrix. The overall experimental architecture of the paper is shown in Figure 1. In summary, the main contributions of this paper are:

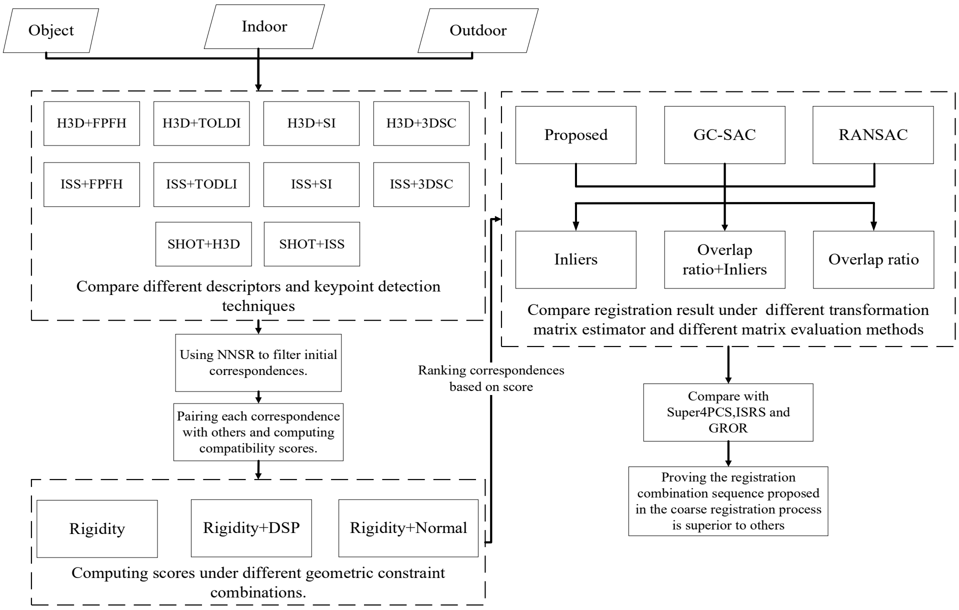

Object
Indoor
Outdoor
H3D+FPFH
H3D+TOLDI
H3D+SI
H3D+3DSC
ISS+FPFH
ISS+TODLI
ISS+SI
ISS+3DSC
SHOT+H3D
SHOT+ISS
Compare different descriptors and keypoint detection techniques
Using NNSR to filter initial correspondences.
Pairing each correspondence with others and computing compatibility scores.
Rigidity
Rigidity+DSP
Rigidity+Normal
Computing scores under different geometric constraint combinations.
Ranking correspondences based on score
Proposed
GC-SAC
RANSAC
Inliers
Overlap ratio+Inliers
Overlap ratio
Compare registration result under different transformation matrix estimator and different matrix evaluation methods
Compare with Super4PCS,ISRS and GROR
Proving the registration combination sequence proposed in the coarse registration process is superior to others

Figure 1. The architecture of the experiment in our article.

(1) We propose an evaluation system based on rigidity constraints and distance of salient point constraints to filter out outliers in the initial correspondences. The rigidity constraint focuses on the geometric relationship between points, while the DSP constraint leverages the context information within the local surface. By combining them, we can better select the top N candidates from the correspondences.

(2) We propose a method to compute the transformation matrix of the source point cloud in reverse by computing the rotation and translation matrices between the source and target points based on the correspondences. Considering two high-quality correspondences from the samples for calculating the transformation matrix allows us to generate matrix with few iterations and also increases the probability of finding the optimal matrix.

(3) To further refine the registration process, we propose an evaluation method based on the overlap ratio and inlier points. This evaluation method allows us to search for the best registration result among the top N candidates in a more comprehensive manner.

2. The Principle of Proposed Registration

First, this paper adopts a keypoint detection method and feature descriptor to generate initial correspondences. Then, we apply the nearest neighbor similarity ratio to filter these initial correspondences and select higher quality ones. Next, within each correspondence, we compute salient points for the source and target points, and combine them pairwise. Based on rigidity and salient points' distance constraint, we select higher quality ones. Next, within each point pair, we compute salient points for the source and target points, and combine them pairwise. Based on rigidity and salient points' distance constraint, we select out correspondences with higher scores. Finally, using the local reference frames between the two points and their centroid positions, we compute the transformation matrix. The best matrix is selected based on the overlap ratio and the number of inlier points. The technical flow of the entire method is shown in Figure 2.

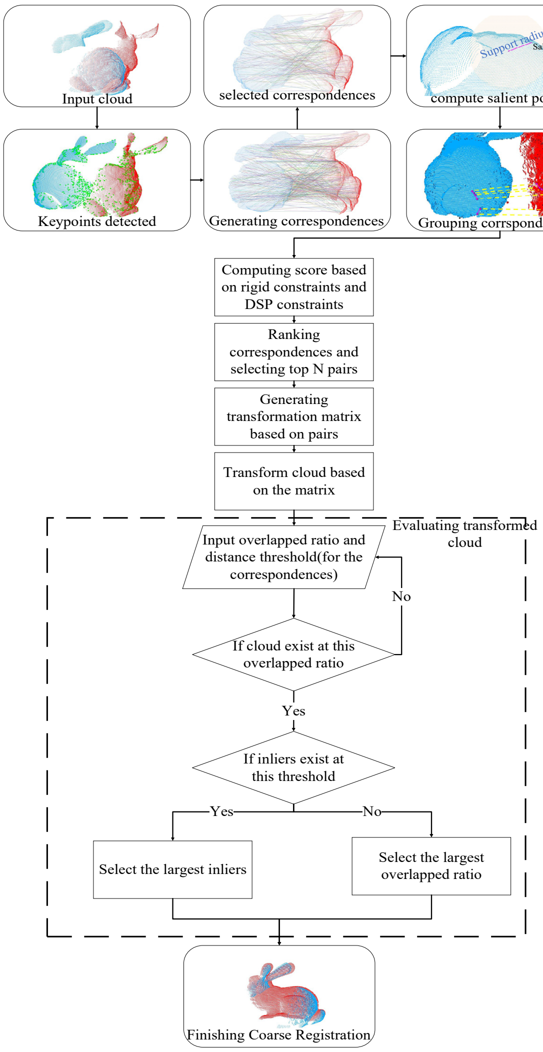

Input cloud
selected correspondences
Support radius
Salient point
compute salient points
Keypoints detected
Generating correspondences
Grouping corrspondences
Computing score based on rigid constraints and DSP constraints
Ranking correspondences and selecting top N pairs
Generating transformation matrix based on pairs
Transform cloud based on the matrix
Evaluating transformed cloud
Input overlapped ratio and distance threshold(for the correspondences)
No
If cloud exist at this overlapped ratio
Yes
If inliers exist at this threshold
Yes
No
Select the largest inliers
Select the largest overlapped ratio
Finishing Coarse Registration

Figure 2. The workflow of the coarse point cloud registration method based on geometric constraints and the two-factors evaluation. The red object represents the target point cloud and the blue one represents the source point cloud. The green points represents keypoins that are detected from the object. In Grouping correspondences, the yellow dotted line represents the correspondences, and the purple line represents the combination of two correspondences.

2.1. Generate Correspondences

First, keypoints are extracted from the source and target point clouds, using keypoints extraction methods such as Harris 3D (H3D) [28] and intrinsic shape signatures (ISS) [29]. Then we combine them with descriptors such as fast point feature histogram (FPFH) [30], 3D shape context descriptor (3DSC) [31], signature of histograms of orientations (SHOT) [32], and spin image (SI) [33] for comparative experiments. These descriptors are used to generate initial correspondences using the similarity score algorithm [6]. Based on the comparative results mentioned in the subsequent experiments, we choose to use the combination of Harris 3D keypoint detection and the local image descriptors obtained from the triplet orthogonal views based on the newly proposed local reference frame (LRF), known as triple orthogonal local depth images (TODLI) [34], for generating initial correspondences.

After obtaining the keypoints using Harris 3D, we generate histograms based on the TOLDI descriptors for both the keypoints in the source point cloud, denoted as p^s, and the keypoints in the target point cloud, denoted as p^t. We calculate the correspondences' differences between the feature values of the histograms and select the keypoint pairs with the smallest differences as the initial correspondences, as defined in (1).

$$c = \operatorname{argmin}\left\|f^s - f^t\right\|_{L_2} \tag{1}$$

In this step, we calculate the feature distance between feature f^s in the source point cloud and feature f^t in the target point cloud. We select the correspondence c with the smallest feature distance as the initial correspondences. Next, we use the NNSR [10] to improve the quality of the correspondences. The NNSR is defined as the ratio of the feature distances between the source point cloud and the target point cloud in each correspondence, as shown in (2). All correspondences are sorted based on this ratio. The ratio can be used to measure the uniqueness and accuracy of the correspondences. Generally, correspondences with a larger feature distance ratio (greater difference between the closest and second closest distances) tend to have higher uniqueness and are assigned higher scores. By applying the NNSR filter, we can further reduce the influence of incorrect matches and noise point pairs, and select higher quality correspondences as the final set of initial correspondences.

$$s_{\text{Ratio}}\,(c) = 1 - \frac{d\left(f^s, f_1^t\right)}{d\left(f^s, f_2^t\right)} \tag{2}$$

The features f_1^s and f_2^t represent the closest and second closest features, respectively, to the feature f^s in the source point cloud p^s. In practical applications, the choice of the appropriate number of correspondences can be based on specific requirements and tasks. Selecting an appropriate number of point pairs helps balance the registration accuracy, computational complexity, and robustness. We will conduct this experiment in the following part to find the appropriate parameter.

2.2. Selecting Correspondences Based on Geometric Constraint

By incorporating various geometric constraints, additional information and constraints can be provided in the point cloud registration process, assisting in the selection of matching point pairs and further enhancing their quality. In this section, we introduce three constraint methods that primarily rely on these two correspondences, c_1 and c_2.

The rigidity constraint [8] is based on the invariance property and determines whether the two points from the source point cloud and the two points from the target point cloud should be considered as the same correspondence based on the difference in their distances. This constraint is represented by (3) and illustrated in Figure 3.

$$\mathcal{S}_{\mathrm{rigidity}} = \left|\mathrm{d}\left(\mathrm{p}_1^{\mathrm{s}}, \mathrm{p}_2^{\mathrm{s}}\right) - \mathrm{d}\left(\mathrm{p}_1^{\mathrm{t}}, \mathrm{p}_2^{\mathrm{t}}\right)\right| \tag{3}$$

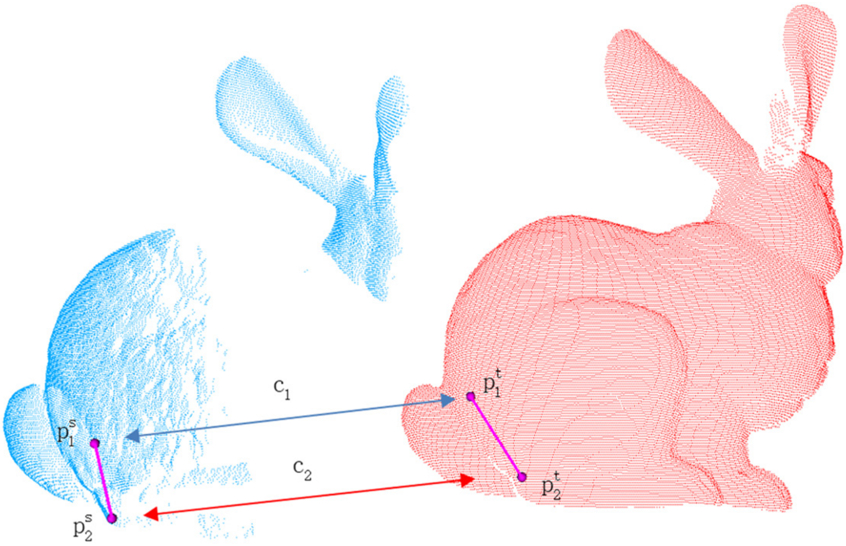


Figure 3. Illustration of a rigidity constraint, consisting of any two correspondences. The red bunny represents the target point cloud and the blue one represents the source point cloud.

The distributions of $\mathrm{d}\left(\mathrm{p}_1^{\mathrm{s}}, \mathrm{p}_2^{\mathrm{s}}\right)$ and $\mathrm{d}\left(\mathrm{p}_1^{\mathrm{t}}, \mathrm{p}_2^{\mathrm{t}}\right)$ represent the Euclidean distances between the points $\mathrm{p}_1^{\mathrm{s}}$ and $\mathrm{p}_2^{\mathrm{s}}$ within the source correspondences, as well as between the points $\mathrm{p}_1^{\mathrm{t}}$ and $\mathrm{p}_2^{\mathrm{t}}$ within the target correspondences. In the process of point cloud registration, the rigidity constraint serves as a fundamental constraint that provides an initial estimation for subsequent optimization and refinement steps. However, the estimation based on the rigidity constraint is typically coarse and can lead to multiple rigid transformations that satisfy the constraint. Therefore, it is necessary to combine other constraints to reduce ambiguity.

The normal constraint [35] is another type of constraint used in point cloud registration. It utilizes the normal vectors associated with the points in the point clouds. After obtaining the normal, the normal constraint compares the angles between the normal vectors of two corresponding points in the source and target point clouds. This is achieved by calculating the cosine values of the angles (using the dot product of the unit normal vectors). The degree of the normal constraint is quantified by computing the absolute difference between the angles, as shown in (4).

$$\mathcal{S}_{\mathrm{normal}} = \left|\mathrm{acos}\left(\mathrm{n}_1^{\mathrm{s}} \cdot \mathrm{n}_2^{\mathrm{s}}\right) - \mathrm{acos}\left(\mathrm{n}_1^{\mathrm{t}} \cdot \mathrm{n}_2^{\mathrm{t}}\right)\right| \tag{4}$$

where n is the normal vector of point p. Both n_1 and n_2 are treated as unit normal vectors. So when the dot product is performed on these vectors, the resulting value is the cosine of the angle between the two vectors. The normal vectors are important features that describe the geometric properties of surfaces. The different normal angles between the source and target point clouds reflect the degree of disparity in surface curvature. Compared to the rigidity constraint, the normal constraint exhibits better robustness when dealing with noise and local variations. Even in the presence of noise or local non-rigid deformations, the normal constraint can still provide useful constraint information. However, the normal constraint only considers the normal angles between local correspondences, and it may not accurately capture the geometric information for cases involving large-scale shape variations or complex geometric structures. Moreover, the uncertainty in normal directions should also be taken into account. Selecting inconsistent normal directions as references can lead to erroneous angle calculations, resulting in misleading constraint results, as shown in Figure 4.

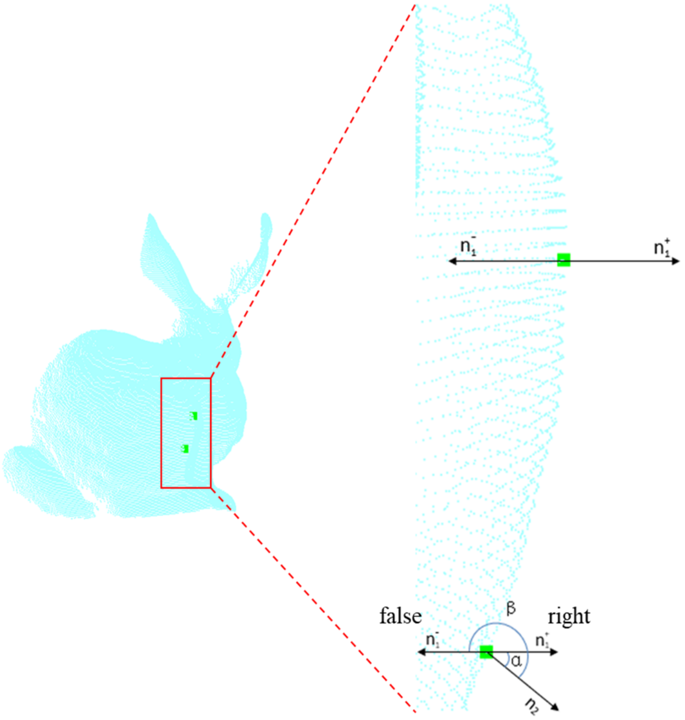


Figure 4. We use the two green points from the blue bunny object as a demonstration of the uncertainty of the normal direction. The uncertainty in normal directions arises when, for example, the direction n_1^+ is the correct normal direction, but n_1^- is mistakenly chosen as the normal direction. In this case, the normal angle increases from α to β, resulting in an incorrect estimation of the normal angle.

The distance of the salient point (DSP) constraint [13] is derived based on the difference in distance between the center point p and its corresponding salient point q^*. The salient point is selected from the boundary region of the local surface, satisfying two conditions: firstly, within this boundary region, the vector pq^* has the longest length; secondly, the vector pq^* has the most similar direction to the normal direction of centroid p. These conditions make the point q^* exhibit saliency characteristics. An illustration of salient points is shown in Figure 5.

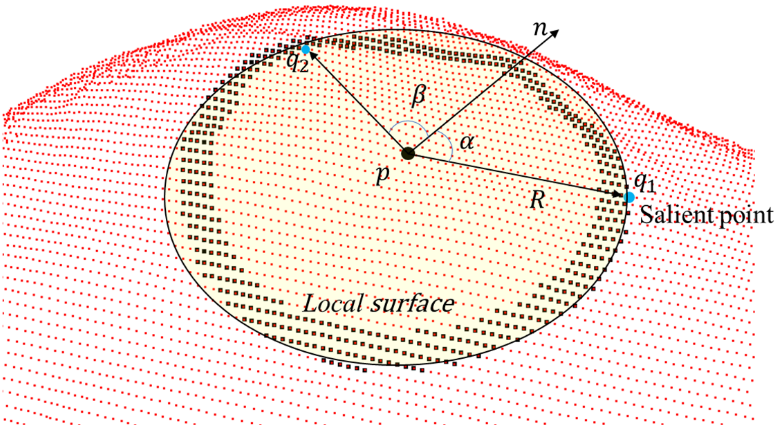


Figure 5. In the computation of salient points, for example, if the absolute value of vector pq_1 is greater than that of pq_2, and the direction of vector pq_1 is more similar to the normal vector n compared to pq_2 ($\cos\alpha > \cos\beta$), we consider q_1 as the salient point. The black dots represent the points on the boundary region, and the blue dot represent the point that qualify as salient point on the boundary region.

As shown in (5), q^* represents a point on the boundary region that satisfies two conditions: the vector pq has the maximum length among all points in the boundary region, and its direction is consistent with the normal vector n of point p within a local surface with a radius of R. These imply that pq aligns to some extent with the tangent plane of the local surface. In this case, pq can be considered as a point on the surface with the maximum saliency.

$$\mathrm{q}^\star = \underset{\mathrm{q}}{\mathrm{argmax}}|\mathrm{qp}\cdot\mathrm{n}| \tag{5}$$

The radius R of the local surface is determined by considering a neighborhood of points around p as shown in (6). Specifically, we select the $\left\lceil \frac{1}{\mathrm{pr}} \right\rceil$ points $\mathrm{p}_{\mathrm{nn_i}}$ in the neighbor of p, denoted as k points. By using the point cloud resolution as a measure of the number of neighboring points, we ensure that the support radius is consistent with the sampling density of the point cloud. This choice of support radius allows for a better representation of the local surface's geometric characteristics.

$$\mathrm{R} = \frac{1}{\mathrm{k}}\left(\sum_{i=1}^{k}\left\|\mathrm{p}-\mathrm{p}_{\mathrm{nn_i}}\right\|\right) \tag{6}$$

After computing the salient points for each point, we apply a similar approach to the rigidity constraint by comparing the distance difference between q_1^* and q_2^*, as shown in (7).

$$\mathcal{S}_{\mathrm{dsp}} = \left|\left\|(\mathrm{q}_1^{\star\mathrm{s}} - \mathrm{p}_2^{\mathrm{s}}\mathrm{p}_1^{\mathrm{s}}) - \mathrm{q}_2^{\star\mathrm{s}}\right\| - \left\|(\mathrm{q}_1^{\star\mathrm{t}} - \mathrm{p}_2^{\mathrm{t}}\mathrm{p}_1^{\mathrm{t}}) - \mathrm{q}_2^{\star\mathrm{t}}\right\|\right| \tag{7}$$

However, in addition to this, we also translate the salient points by a vector $\mathrm{p}_2\mathrm{p}_1$. By aligning the salient points with the center of p_2, we eliminate the effect of spatial distance variation between p_1 and p_2. This allows for a more accurate comparison of local structural differences in the two correspondences, taking into account not only the overall rigid transformation, but also the local variations, as illustrated in Figure 6a.

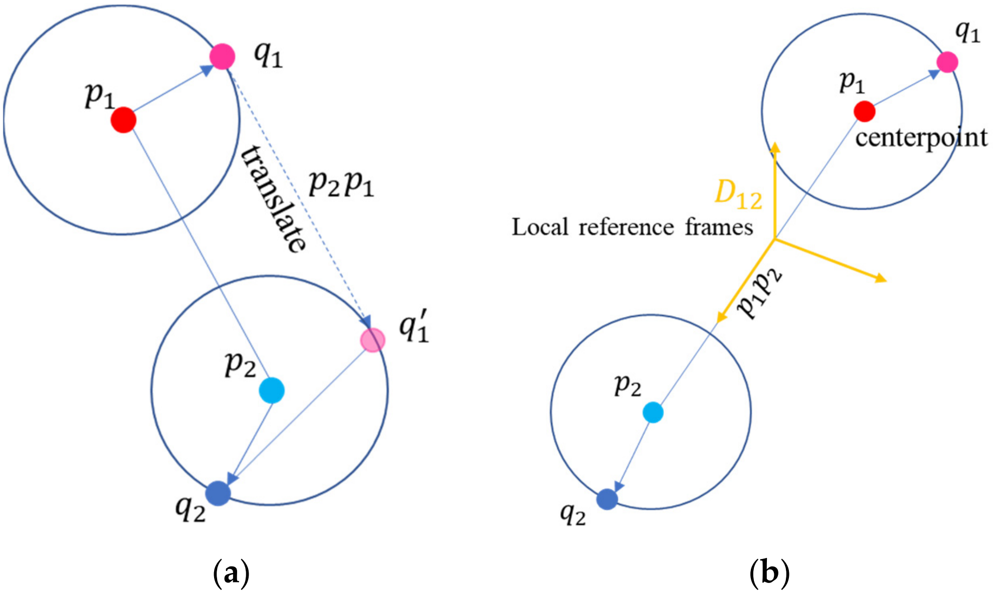


Figure 6. The red and blue points in the figure represent two source points from a grouping correspondences. The pink and dark blue points correspond to their salient points. Illustration of Salient points' distance constraint (**a**); generating the transformation matrix with two correspondences (**b**) based on the centroid p an d the salient point q to establish a local reference frame (also demonstrated as the rotation pose in the transformation matrix).

Afterwards, we combine the rigidity constraint and the distance of salient points constraint. These two constraints are eventually combined as S_{both} to compute the score of the correspondences c_1 and c_2 given by (8).

$$s_{both}\left(c_1, c_2\right) = \exp\left(-\frac{\mathcal{S}_{rigidity}}{(a \cdot pr)^2} - \frac{\mathcal{S}_{dsp}}{(b \cdot pr)^2}\right) \tag{8}$$

To weight the scores of each constraint, we use an exponential function. This allows the rigidity constraint ($S_{rigidity}$) and the DSP constraint (S_{dsp}) to have a more significant impact on decreasing the overall score if their values are higher, thus favoring the exclusion of correspondences that do not satisfy the constraints. In this formulation, $a * pr$ and $b * pr$ serve as distance thresholds for $S_{rigidity}$ and S_{dsp}, respectively. When the constraints of correspondences are below these thresholds, it indicates that the geometric differences between the two correspondences are within an acceptable range, and the score will be maintained at a relatively high level. By setting appropriate values for a and b, we can achieve a desirable distance threshold. Here, pr represents the point cloud resolution, and it is defined by the Formula (9).

$$pr = \frac{1}{|\mathcal{P}|}\sum_{p \in \mathcal{P}} \frac{1}{10}\left(\sum_{i=1}^{10}\left\|p - p_{nn_i}\right\|\right) \tag{9}$$

We calculate the average distance of the 10 nearest neighbor points p_{nn_i} around each point p in the point cloud $\mathcal{P}$. Then we compute the average distance of all points in the cloud. After that, we select the top-ranked correspondence combinations and calculate the local reference frame for the two correspondences c_1 and c_2, as shown in (10).

$$D_{12} = \left[\frac{p_1p_2 \times \left(p_1q_1^* + p_2q_2^*\right)}{\left\|p_1p_2 \times \left(p_1q_1^* + p_2q_2^*\right)\right\|}\ \frac{p_1p_2}{\|p_1p_2\|}\ \frac{p_1p_2 \times \left(p_1q_1^* + p_2q_2^*\right) \times p_1p_2}{\left\|p_1p_2 \times \left(p_1q_1^* + p_2q_2^*\right) \times p_1p_2\right\|}\right] \tag{10}$$

Here, we establish the first axis of the local reference frame based on the plane formed by $p_1q_1^* + p_2q_2^*$ and p_1p_2. $P_1q_1^* + p_2q_2^*$ to integrate the information of the two salient points. Additionally, p_1p_2 serves as the second axis. We then construct the third axis, which is perpendicular to the plane formed by the first two axes, using the $p_1p_2 \times \left(p_1q_1^* + p_2q_2^*\right) \times p_1p_2$ vector. These three vectors form an orthogonal reference coordinate system, as shown in Figure 6b.

Finally, we can calculate the rotation and translation matrix based on the formula (11) proposed by Quan, S., and J. Yang [13].

$$H = \begin{bmatrix} D_{12}^t & \frac{p_1^t + p_2^t}{2} \\ 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} D_{12}^s & \frac{p_1^s + p_2^s}{2} \\ 0 & 1 \end{bmatrix} \tag{11}$$

In the formula, $H_t = \begin{bmatrix} D_{12}^t & \frac{p_1^t + p_2^t}{2} \\ 0 & 1 \end{bmatrix}$ and $H_s = \begin{bmatrix} D_{12}^s & \frac{p_1^s + p_2^s}{2} \\ 0 & 1 \end{bmatrix}$ represent the matrices formed by the center positions and rotation poses of two points in the source and target point clouds, respectively. However, in the formula $H_t \times H = H_s$, the transformation matrix is applied to the target point cloud, returning it to the original coordinate system. We need to modify the formula to $H \times H_s = H_t$ so that the transformation matrix acts on the source point cloud. The modified transformation matrix formula should be as follows (12):

$$H = \begin{bmatrix} D_{12}^t & \frac{p_1^t + p_2^t}{2} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} D_{12}^s & \frac{p_1^s + p_2^s}{2} \\ 0 & 1 \end{bmatrix}^{-1}. \tag{12}$$

2.3. Matrix Evaluation

After generating the transformation matrix for the highly ranked correspondences, we need to evaluate the coarse registration results. We use two metrics to evaluate the results: overlap ratio and the number of inlier points. The overlap ratio can be calculated using two methods: KD-tree and octree. The KD-tree employs a nearest neighbor search algorithm to find the closest points to the query point within a given radius. On the other hand, the octree uses a voxelization approach to check for the existence of point samples within each voxel. Based on a comparative experiment between these two methods (as shown in Figure 7), we adopt the octree method to calculate the overlap ratio.

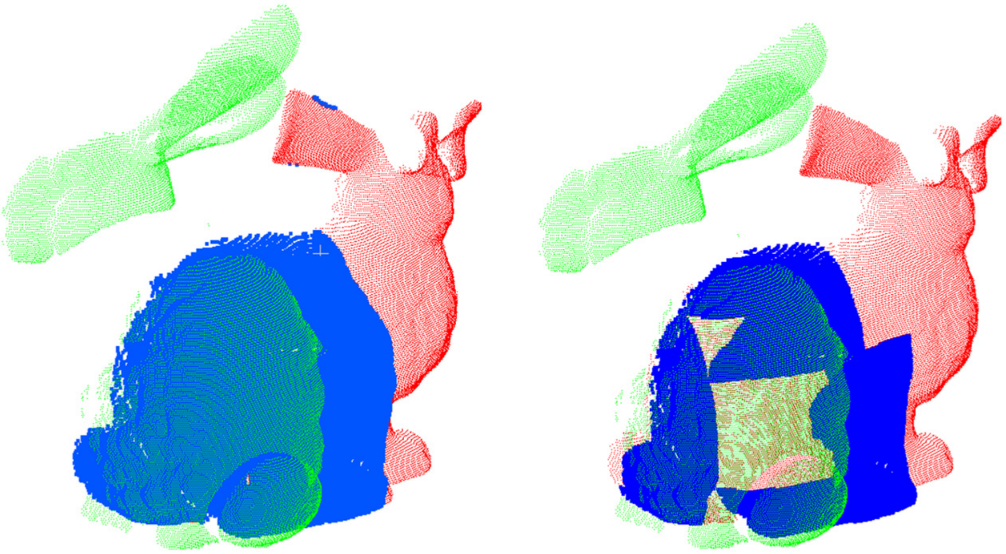


Figure 7. The green bunny model and the red bunny model represent unregistered point clouds. The blue areas represent their overlapping areas. In the left figure, the overlap ratio calculated using the kd-tree method is 60%, while in the right figure, the overlap ratio calculated using the octree method is 44%.

From the images, it can be observed that the octree method is able to more accurately identify the overlapping regions. Additionally, the octree method also demonstrates better search efficiency compared to the kd-tree method, which is beneficial for evaluating a large number of matrices and complex scenes. As for the specific steps of evaluating, firstly, we select the top-ranked registration results based on the overlap ratio threshold. Then, using the number of inlier points within the distance threshold to choose the best transformation matrix within these top-ranked results. We employ two metrics for evaluation for multiple reasons. Firstly, the overlap ratio considers the overall registration accuracy and reflects the alignment of the two point clouds on a global scale. Secondly, the number of inlier points takes into account the local quality of the registration results. Correct correspondences demonstrate consistency in geometric shape and topological structure within the local region. A higher number of inlier points signifies more precise and stable registration results within the overlapping region. Therefore, by considering both the overlap ratio and the number of inlier points, we can comprehensively evaluate the accuracy and completeness of the registration results.

Finally, we convert these main steps into Algorithm 1.

Algorithm 1 Proposed registration method

```
Require: Point clouds $p^s$ and $p^s$, and correspondence set $C$;
Ensure: Generate the best transformation matrix;
Using TOLDIs descriptors and H3D keypoint detection to generate feature Histogram and
using Equation (1) to generate initial correspondences;
Using Equation (2) to select the number of $C$
  $C = 200$;
Evaluate each correspondence based on the Equation (8)
  in $C$ using Equation (8);
Ranking samples based on the compatibility score. Then we select out top $N$ correspondences
as candidates
  $N = 300$;
while $i < N$ do
Compute transformation $H$ based on $(c_1, c_2)_i$ using Equation (11);
Calculate inlier number $I_i$ and overlap ratio $O_i$ for each $H_i$;
$i = i + 1$;
end while
Setting threshold for overlap ratio
  $O_T = 0.5$; Then we use $H_T$ to store the $H_i$ that fit $O_T$
while $i < N$ do
  If $O_i > O_T$ then
    $H_T \leftarrow H_i$
  else Re-enter the ovelap ratio until there is a point cloud exist
  end if
  $i = i + 1$;
end while
Select the $H$ from $H_T$ with highest $I_i$ as the optimal matrix.
```

3. Experiments and Discussion

This section focuses on verifying the accuracy of the point cloud coarse registration method proposed by this article. The entire experiment was conducted using the Point Cloud Library (PCL 1.12.1) with C++ programming language on a PC with an i7-9700 processor and 16 GB of RAM.

3.1. Experimental Setup

The experiment utilized various datasets, including Bunny, Dragon, and Armadillo from the Stanford dataset; kitchen and indoor scenes scanned by Princeton University; “Iqmulus & TerraMobilita Contest” [36] dataset is an urban environment in Paris, acquired by the French National Mapping Agency through mobile laser scanning (MLS); and the Taoist Zhenwu Temple in Rong County, Guangxi, collected by our team, as registration data (Table 1). These datasets exhibit different point cloud densities, overlap ratios, and application scenarios. By incorporating the diversity of these datasets, we can evaluate the practical applicability of the proposed method.

Table 1. Experimental data.

No.	Data	Source Points	Target Points	Overlap Ratio	Resolution	Scenario
1	Bunny	30,379	40,251	0.60304	0.001	Object
2	Dragon	41,841	22,092	0.36715	0.00099	Object
3	Armadillo	26,941	25,570	0.81912	0.001	Object
4	redKitchen	258,342	268,977	0.47331	0.0081	Indoor
5	Home	425,577	373,295	0.74085	0.008	Indoor
6	Paris	372,620	204,128	0.85941	0.73	Outdoor
7	Zhenwu Temple	2,273,238	2146,665	0.84996	0.018	Indoor

In terms of the evaluation criteria, the root mean square error (RMSE) [37] method is commonly used as a standard in various fields. We first calculated the distance error between corresponding points using the Formula (13).

$$\epsilon_{\mathrm{p}}(\mathrm{p}^{\mathrm{s}}, \mathrm{p}^{\mathrm{t}}) = \left\| \mathrm{R}\mathrm{p}^{\mathrm{s}} + \mathrm{t} - \mathrm{p}^{\mathrm{t}} \right\| \tag{13}$$

where R represents the rotation matrix and t represents the translation matrix. Additionally, p^{s} and p^{t}, respectively, represent the source point and the target point from the true correspondence. RMSE is defined as follows (14):

$$\mathrm{RMSE} = \sum_{(\mathrm{p}^{\mathrm{s}}, \mathrm{p}') \in \mathcal{C}_{\mathrm{gt}}} \frac{\epsilon_{\mathrm{p}}(\mathrm{p}^{\mathrm{s}}, \mathrm{p}^{\mathrm{t}})}{|\mathcal{C}_{\mathrm{gt}}|}. \tag{14}$$

$\mathcal{C}_{\mathrm{gt}}$ refers to the ground truth correspondences within the distance threshold. These correspondences are determined by comparing the distances between correspondences in correctly registered point cloud data. Specifically, the correspondences whose distances meet the distance threshold are considered as the ground truth correspondences. However, relying solely on the RMSE value does not effectively reflect the quality of point cloud registration. Some locally optimal matches can result in low RMSE values. The RMSE value becomes meaningful only when both point clouds are accurately aligned as a whole. Precision [38] was added to make the evaluation more comprehensive.

$$\mathrm{Precision} = \frac{|\mathcal{C}_{\mathrm{inlier}}^{\mathrm{corret}}|}{|\mathcal{C}_{\mathrm{inlier}}|} \tag{15}$$

$\mathcal{C}_{inlier}$ refers to the correspondences that satisfy the distance threshold after the registration process. Through multiple experiments, the distance threshold is typically set around 20*pr. Correspondences that meet this threshold ensure that the source and target point clouds are geometrically close. $\mathcal{C}_{inlier}^{correct}$ specifically refers to the correct correspondences that are in $\mathcal{C}_{inlier}$ and also meet the $\mathcal{C}_{gt}$ standard. These correspondences are originally correct and remain accurate after registration process.

3.2. Analysis of Proposed Method

First, we investigate the influence of two key parameters on the registration accuracy of the method: the number of selected correspondences after NNSR selecting and the number of optimal transformation matrix. The experiment is conducted on the BUNNY090 and BUNNY180 datasets for registration. We use the TOLDI descriptor and Harris3D keypoint detection as the method for generating correspondences.

According to the results shown in Figure 8, the precision and RMSE tend to stabilize when the NNSR method selects approximately 300 correspondences. After considering the trade-off between computational efficiency and registration accuracy, we choose 300 correspondences as the experimental parameter. The 300 correspondences perform well in terms of RMSE and precision, and provide a sufficient sample size for accuracy calculation, allowing for a more comprehensive evaluation of the practical effect of point cloud registration.

Based on the results shown in Figure 9, different transformation matrices yield consistent registration results ranging from 100 to 600, except for 400 and 250 correspondences. Considering the number of provided transformation matrices for evaluation and computational efficiency, we choose 300 hypothesized transformation matrices as the optimal parameter. Having too many transformation matrices can increase the computation time for metrics such as overlap ratio and inlier count, while having too few may result in missing some high-quality transformation matrices.

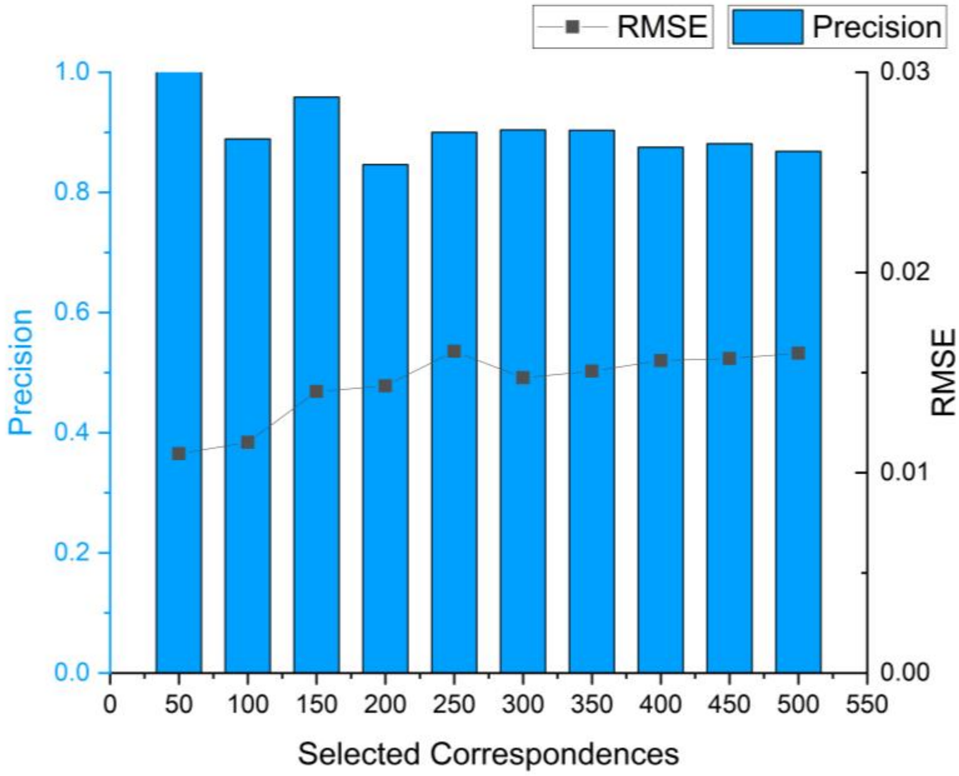


Figure 8. The impact of different numbers of correspondences selected by the NNSR method on registration accuracy.

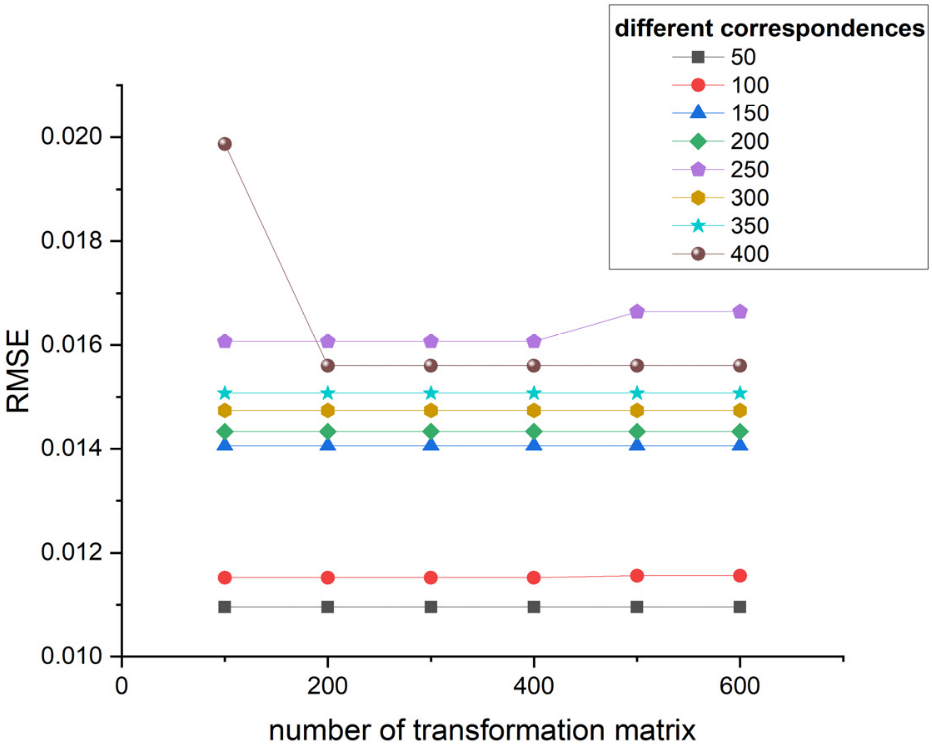


Figure 9. Registration results of different hypothetical transformation matrices corresponding to different correspondences.

Figure 10 illustrates the influence of different transformation matrices on RMSE and precision in the case of 300 correspondences. From the graph, it can be observed that the accuracy results remain consistent across 100 to 800 transformation matrices. This indicates the uniqueness of the optimal matrix within this number of correspondences.

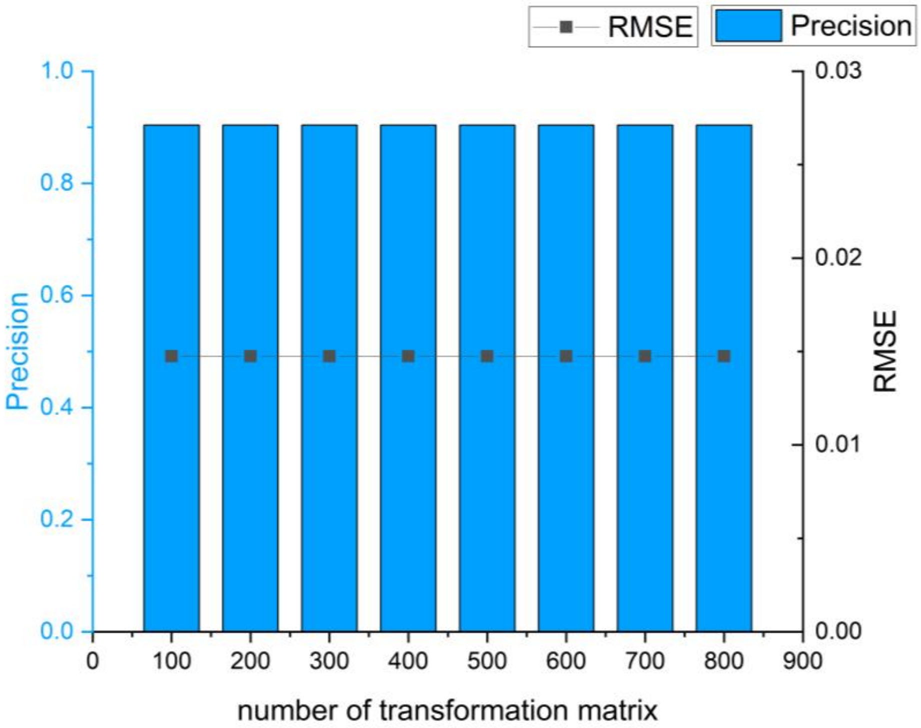


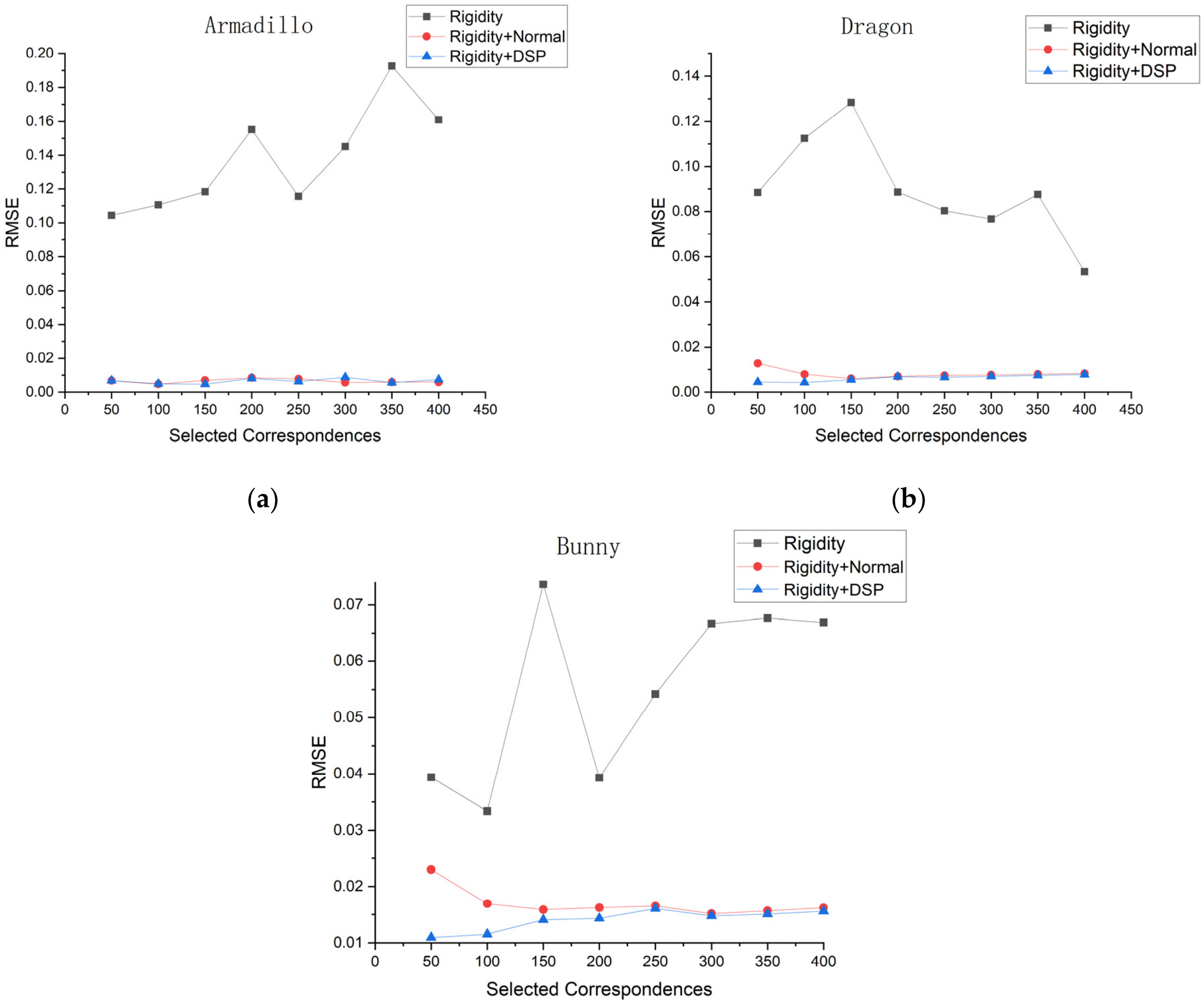
Figure 10. Influence of different hypothetical transformation matrices on registration accuracy under 300 correspondences.

In determining the settings for the distance threshold of the inliers and overlap ratio parameters, based on multiple experiments, we determined to set the distance threshold in the range of 15*pr to 20*pr. This range allows us to assess the proximity between point pairs. The overlap ratio is also a parameter used to evaluate registration accuracy. We utilize the pre-registration overlap ratio as a threshold, as shown in Table 1. This approach helps eliminate misalignments introduced during the registration process and visually showcases the improvements achieved in numbers after registration.

Subsequently, to validate the superiority of combining rigidity constraints with distance of salient point constraints, we conducted tests on three different approaches: rigidity constraint with distance of salient point constraint, rigidity constraint only, and rigidity constraint with normal constraint. We conducted tests on the RMSE values of different object under various geometric constraints (as shown in Figure 11). The registration results of the three different objects clearly indicate that the registration accuracy of the two geometric constraints is superior to using only a single rigidity constraint. By adding the normal constraint, we ensure the similarity in the normal direction of correspondences. Additionally, the salient point constraint utilizes local surface information and provides effective assistance in situations where the rigidity constraint may be ambiguous.

The RMSR value of the Rigidity + DSP constraint can be observed to be superior to the Rigidity + Normal constraint at 50 to 200 correspondences. Both constraint exhibit similar RMSE when the number of correspondences reaches 300. However, there is still an average difference of 0.0005 between them in the RMSE value, proving that the Rigidity + DSP constraint is still superior. As shown in , overall, the combination of the rigidity constraint with the distance of salient point constraint yields better registration results.

In the case of the Bunny, the RMSR value of the Rigidity + DSP constraint can be observed to be superior to the Rigidity + Normal constraint at 50 to 200 correspondences as shown in Figure 12. Both constraints exhibit similar RMSE when the number of correspondences reaches 300. However, there is still an average difference of 0.0005 between them in the RMSE value, proving that the Rigidity + DSP constraint is still superior. The trends of RMSE value in the Dragon are similar to those of the Bunny, as shown in Figure 11b, while in the Armadillo(Figure 11c), both methods exhibit a nearly constant trend. Overall, the registration performance of Rigidity + DSP is superior to Rigidity + Normal when the number of correspondences is small.



(**a**)

(**b**)

(**c**)

Figure 11. (**a**–**c**) respectively represents the registration accuracy of Armadillo, Dragon, and Bunny models respectively under different geometric constraints.

Then, we compared the registration results using different evaluation methods. We conducted tests on the Bunny model and compared the performance of these methods, as shown in Figure 13 and Table 2. These evaluation methods include using the maximum number of inliers and the highest overlap ratio for assessment. We can observe that the method combining inliers and overlap ratio consistently produces the best registration matrix overall. The method based on overlap ratio reaches a similar performance to the inliers and overlap ratio method after generating approximately 200 correspondences. On the other hand, the method based on the maximum number of inliers shows more fluctuations in its registration results and only catches up with the inliers + overlap ratio method when the number of correspondences exceeds 300. In conclusion, among these three evaluation methods, combining the two criteria leads to better registration results.

The registration results of Bunny using different evaluation methods are shown in Figure 14. We selected a range of 50 to 250 correspondences for comparison. It can be observed that, overall, the method combining inliers and overlap consistently achieves good registration results at any number of point pairs.

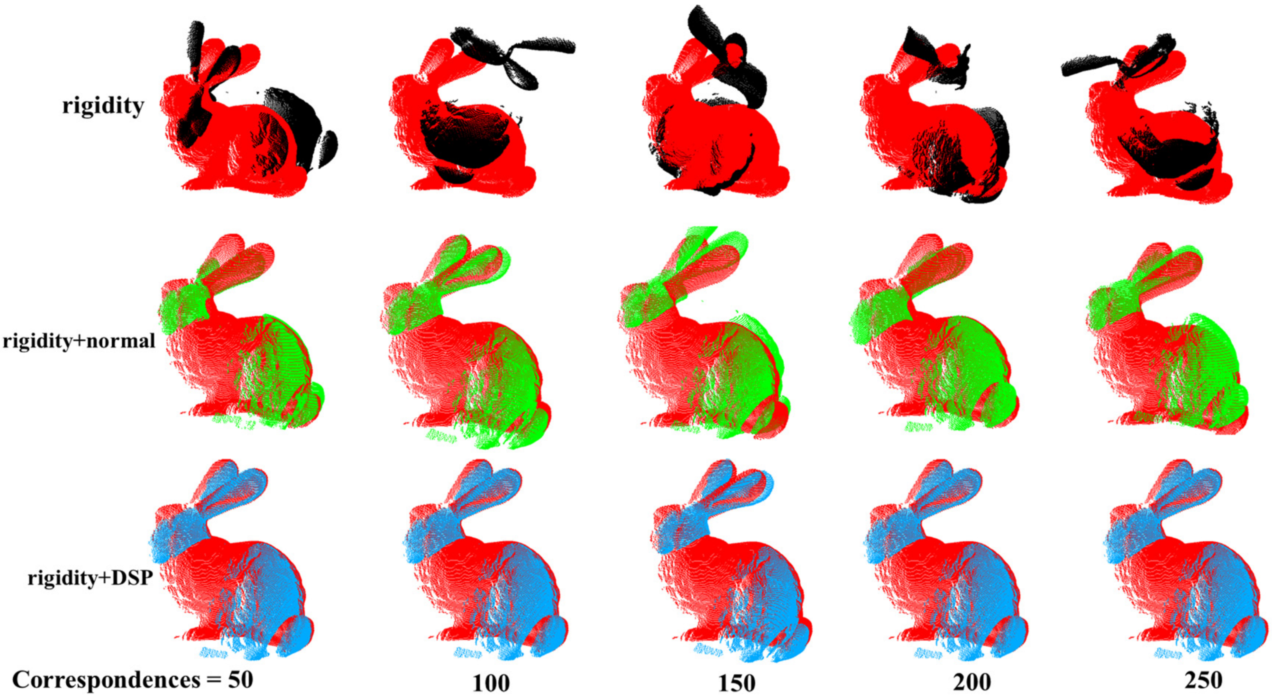


Figure 12. The red bunny represents the target point cloud. The black bunny represents registration under Rigidity constraints. Green one represents registration under the Rigidity + Normal constraint. The blue one represents registration under the constraints of Rigidity + DSP. Figure shows registration result under different geometric constraints.

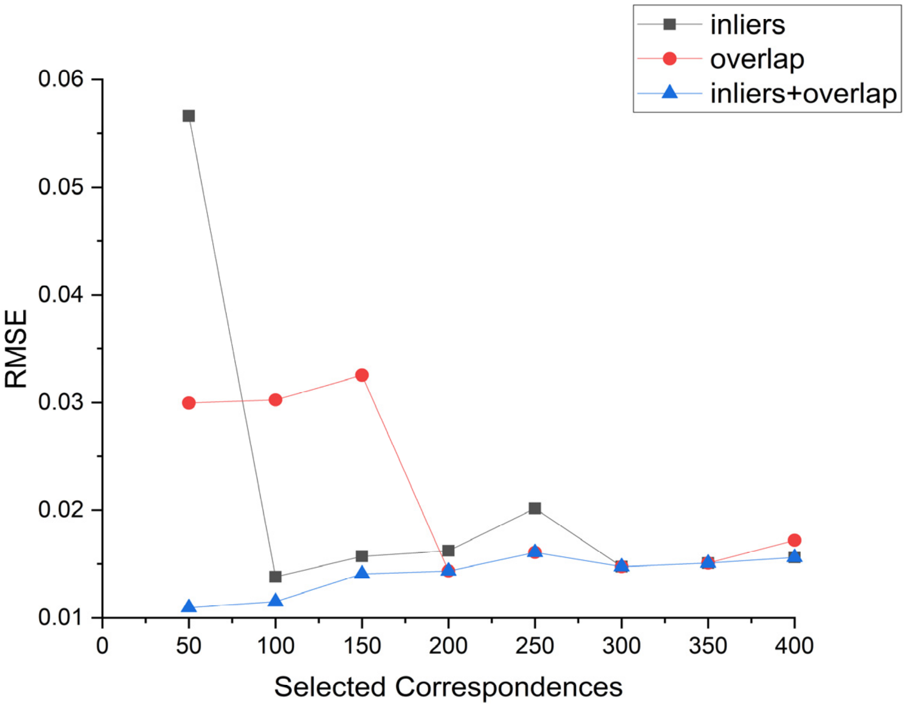


Figure 13. Registration accuracy under different evaluation standards.

Table 2. RMSE result of three evaluation methods.

Selected Correspondences	Inliers	Overlap Ratio	Inliers + Overlap
50	0.05663	0.02998	0.01096
100	0.0157	0.03027	0.01152
150	0.01622	0.03254	0.01406
200	0.02018	0.01433	0.01433
250	0.01474	0.01607	0.01607
300	0.0667	0.01474	0.01474
350	0.01507	0.01507	0.01507
400	0.0156	0.0172	0.0156

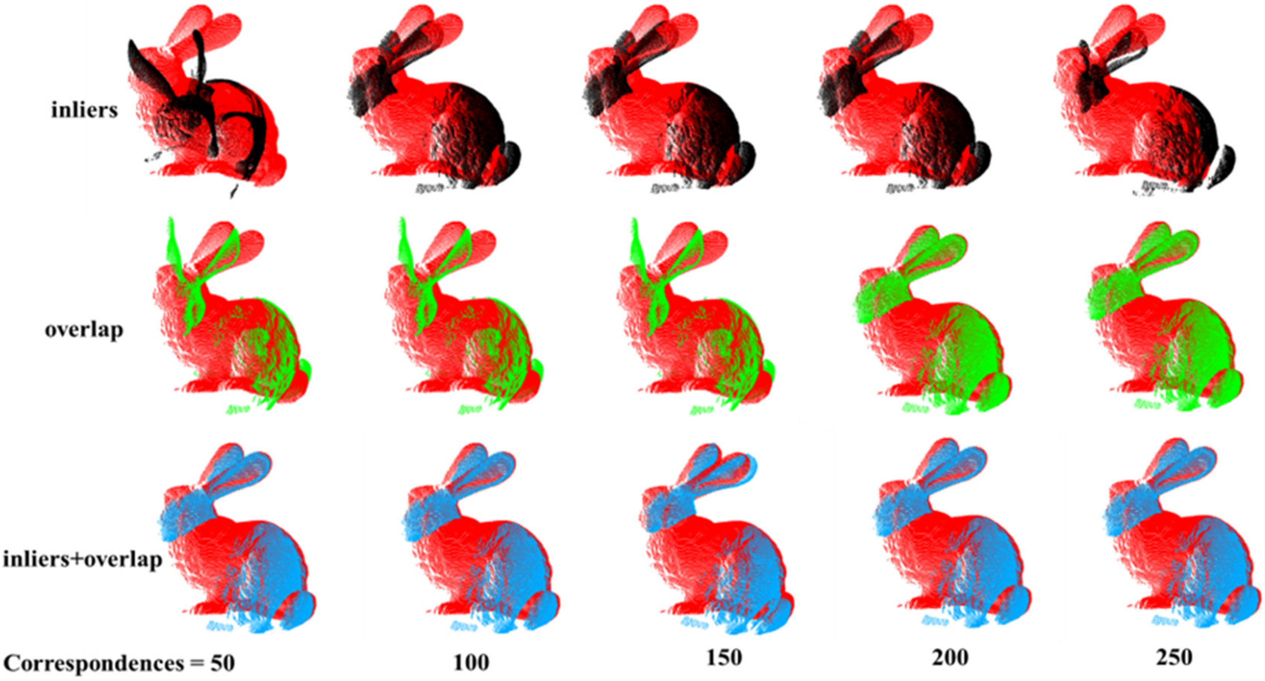


Figure 14. The black bunny represents registration under inliers evaluation criteria. Green one represents registration under the overlap evaluation criteria. The blue one represents registration under the inliers + overlap evaluation criteria. Figure shows registration result under different evaluation criteria.

Finally, we conducted tests on different transformation matrix calculation methods, and the results are shown in Figure 15 and Table 3. Firstly, we compared the proposed method with the GC-SAC method, which is using a different matrix calculation formula. Additionally, it can be seen that the proposed method outperforms the GC-SAC method in terms of registration accuracy. Then, we adopted the RANSAC method to remove the outlier proposed by X. L. [39], and combined it with singular value decomposition (SVD) to calculate the optimal transformation matrix. The results show that in the registration results with 50 to 250 selected correspondences, the proposed method is superior to the RANSAC method overall. In the range of 300 to 400 selected correspondences, the proposed method has a slightly higher average RMSE value compared to the RANSAC method, with a difference of 0.00018. However, the proposed registration method still achieves good results overall, as shown in Figure 16.

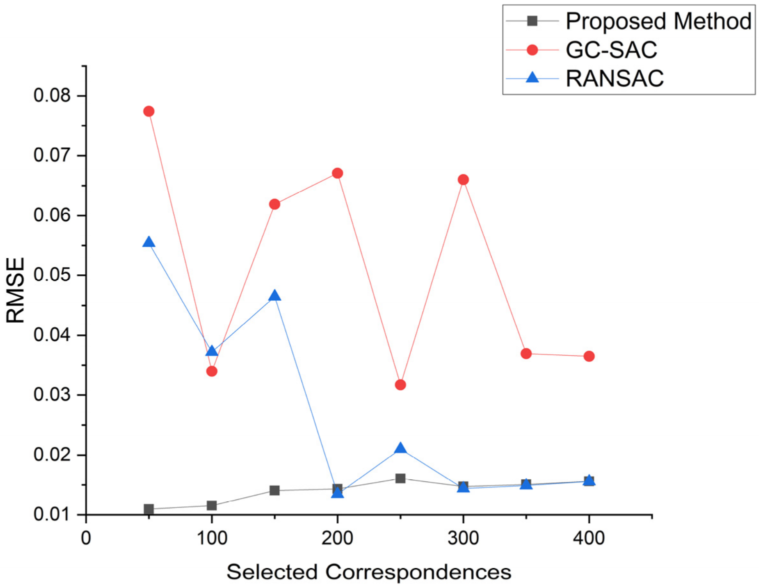


Figure 15. Registration accuracy under different transformation matrix estimators.

Table 3. RMSE result of three transformation matrix estimators.

Selected Correspondences	Proposed	GC-SAC	RANSAC
50	0.01096	0.07739	0.05538
100	0.01152	0.03398	0.0372
150	0.01406	0.06191	0.04648
200	0.01433	0.06711	0.01347
250	0.01607	0.03172	0.02102
300	0.01474	0.06604	0.0144
350	0.01507	0.03694	0.0149
400	0.0156	0.03648	0.01559

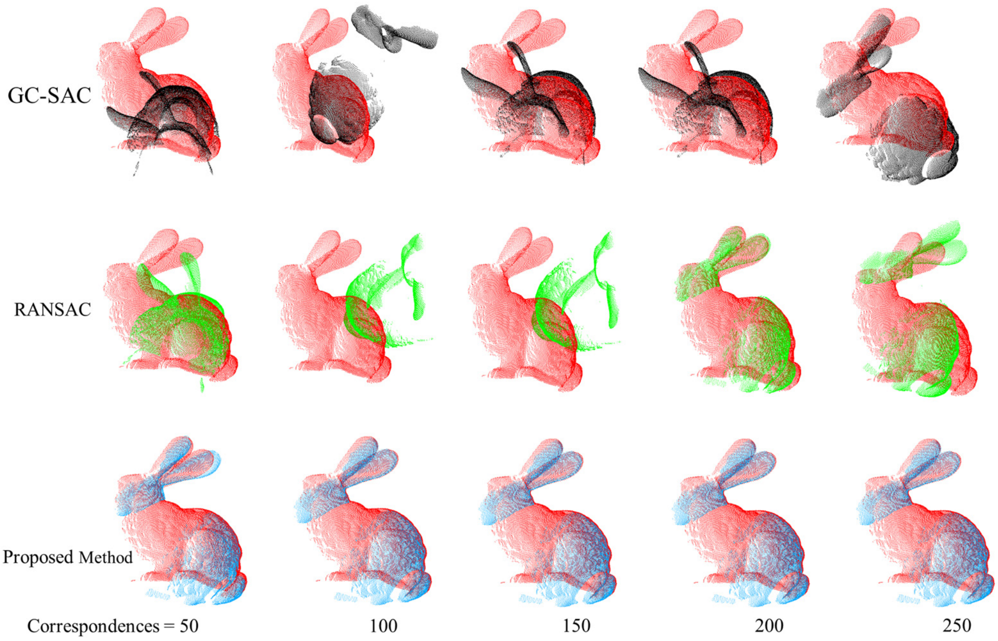


Figure 16. The black bunny represents registration under GC-SAC estimator. Green one represents registration under the RANSAC estimator. The blue one represents registration under our estimator. Figure shows registration result under different transformation matrix estimators.

After confirming the entire registration process, we compared the entire registration process of the proposed method, along with some of the feature-matching registration methods such as IRIS [21], GROR [22], and Super 4PCS [40]. The results are present in Figures 17 and 18. We can observe that in the comparison of three different point cloud objects, the proposed method outperforms the other three methods in terms of RMSE value and actual registration performance. Additionally, the proposed method is similar to Super 4PCS, which also utilizes geometric constraints such as limiting distance range and angle to find the optimal identical four-point matches. However, the proposed method has a lower computational time compared to Super4PCS.

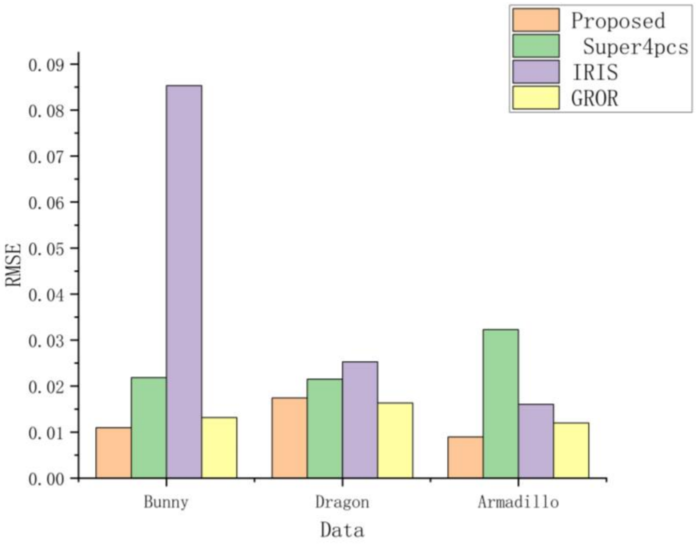


Figure 17. Registration accuracy under different registration methods.

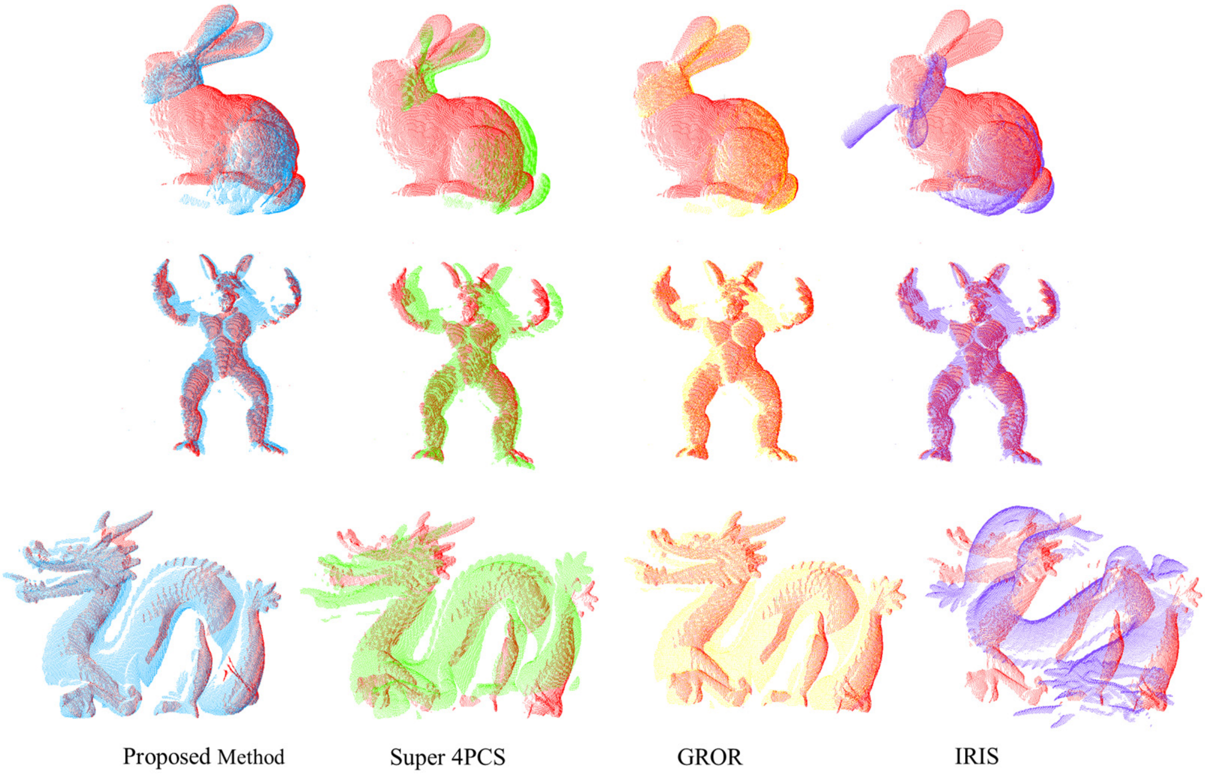


Figure 18. Blue, green, yellow and purple respectively represents our proposed method, Super4PCS, GROR and IRIS. Figure shows Registration results under different registration methods.

3.3. Proposed Method in Practice

Based on the BUNNY090 and BUNNY180 datasets, we evaluated the registration performance of the method using different descriptors and keypoint detection methods. The keypoint detection methods included Harris 3D (H3D) and intrinsic shape signatures (ISS). The descriptors included fast point feature histograms (FPFH), 3D shape context descriptor (3DSC), signature of histograms of orientations (SHOT), spin image (SI), and TOLDIs. We assessed the accuracy of the registered point clouds using different descriptors and keypoint detection methods (Figure 19), as well as the visual quality of the registration results (Figure 20). Detailed results after registration are shown in Table 4.

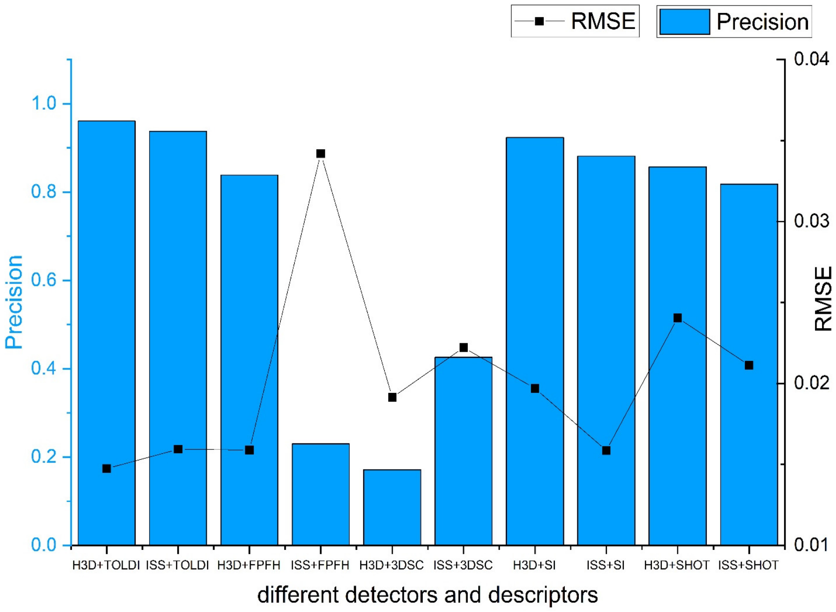


Figure 19. Different combinations of descriptors and keypoint detection methods on registration accuracy.

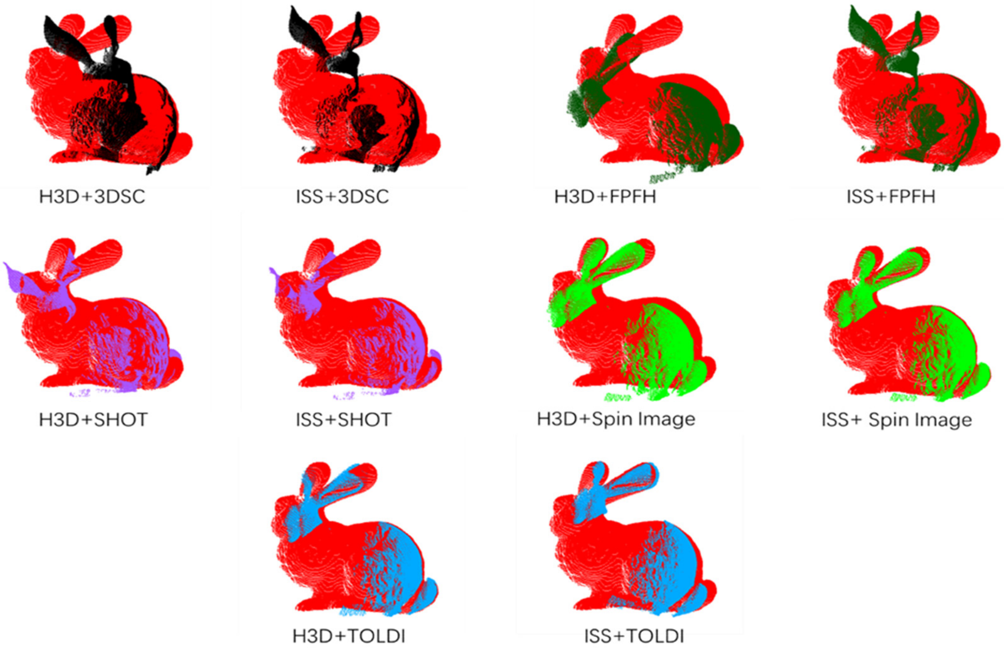


Figure 20. The different colors represent the registration results under different descriptors. Black, dark green, purple, green and blue respectively represents 3DSC, FPFH, SHOT, Spin Image and TOLDI descriptor. Figure shows different combinations of descriptors and keypoint detection methods on registration results.

Table 4. Different descriptors and keypoint detections combined with the proposed method.

Method	RMSE	Precision
TOLDI + H3D	0.014741	0.96
TOLDI + iss	0.01594	0.94
Fpfh + H3D	0.01590	0.84
Fpfh + iss	0.03420	0.23
SI + H3D	0.0197	0.92
SI + iss	0.01586	0.88
3DSC + H3D	0.01914	0.17
3DSC + iss	0.02222	0.43
SHOT + H3D	0.02405	0.8
SHOT + ISS	0.02113	0.82

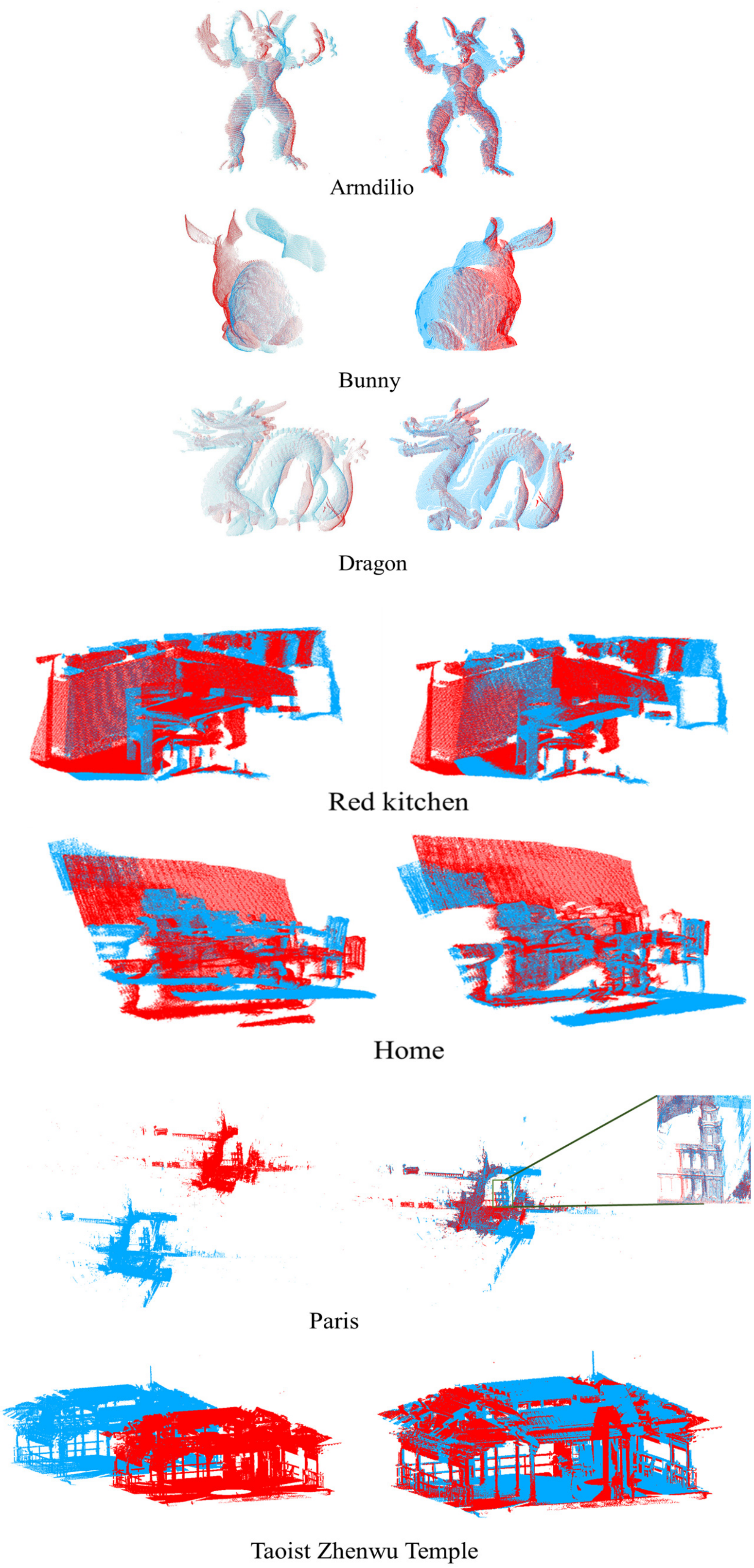
Observing Figures 19 and 20, it can be noted that under the constraints of rigidity and salient points' distance, most of the methods achieve good actual registration results. The RMSE values are maintained around 0.01 to 0.02, and the precision also exceeds 80%. However, the 3DSC descriptor, in combination with both keypoint detection methods, and the FPFH descriptor with the ISS keypoint detection method, did not meet the expected standards in terms of actual registration results.

In the comparison between the combinations of H3D and ISS methods with different descriptors, it can be observed that the registration results with H3D keypoint detection are slightly better than those with ISS keypoint detection. This is because our experimental dataset, Bunny, is more sensitive to corner point features such as the ears and nose of the rabbit, and the Harris 3D method is more suitable for capturing these corner point features. On the other hand, the ISS algorithm focuses on capturing more comprehensive keypoint information of the model, including curvature and normal changes. From the experimental results, it can be concluded that the corner point features of the rabbit have more distinctive characteristics compared to its ISS keypoints.

In terms of descriptors, TOLDI utilizes projections in three orthogonal directions, allowing it to capture local shape features of the point cloud data in different directions. This enables TOLDI to capture more detailed and local structural information in multiple dimensions. This is the reason why TOLDI performs well among all the descriptor methods. The spin image descriptor achieves the second best performance. Since the BUNNY090 and BUNNY180 datasets are obtained from different viewpoints, the spin image descriptor, which calculates rotational projection histograms on the point cloud, can describe local geometric features and counteract noise and inconsistencies between local point clouds through rotational invariance. As a result, it demonstrates good matching performance on point cloud data acquired by rotating at different angles.

The SHOT descriptor has relatively high dimensions, typically around 352 dimensions, while the FPFH descriptor has relatively low dimensions, typically around 33 dimensions. Therefore, the registration results achieved by the SHOT descriptor are superior to the FPFH descriptor. The 3DSC descriptor encodes the geometric relationships between points on a spherical surface and their neighboring points, providing a more comprehensive representation of the overall shape of the point cloud. On the other hand, the FPFH descriptor focuses on the relative angular changes between the neighboring points' normal changes and is suitable for surfaces with significant normal variations or objects with edge features. For the Bunny model, which emphasizes local features, the FPFH descriptor slightly outperforms the 3DSC descriptor.

To test the robustness of proposed registration method on different data types, we conducted experiments on seven datasets: Bunny, Dragon, Armadillo, RedKitchen, Home, Paris, and the Taoist Zhenwu Temple. The Taoist Zhenwu Temple dataset was acquired using the Riegl VZ-1000 3D laser scanner in Rong County, Yulin City, Guangxi, China. It can be observed from Table 5 that the precision of the registrations using the geometric constraints and comprehensive evaluation method remain above 80%. The RMSE values are also below 15pr. Even for complex indoor scenes such as the Taoist Zhenwu Temple, the RMSE is around 5pr. The actual registration results for each dataset are shown in Figure 21.


Armdilio

Bunny

Dragon

Red kitchen

Home

Paris

Taoist Zhenwu Temple

Figure 21. The coarse registration results of each data based on geometric constraints and comprehensive evaluation, the left is before registration and the right is after registration. Blue represents source point cloud and red represent target point cloud.

Table 5. Registration results for different types of data.

Data	RMSE	Precision
Bunny	0.01474	0.90385
Dragon	0.01742	0.8932
Armadillo	0.00894	1
redKitchen	0.02249	0.84906
Home	0.04516	0.827
Paris	0.52034	0.948
Taoist Zhenwu Temple	0.135652251	0.850746269

Figure 22 displays the local details of the registration for the Taoist Zhenwu Temple. Even for complex historical architectural structures and with voxel filtering applied to the raw data, the proposed method can achieve structural registration. This verifies that the registration method is capable of providing high-quality registration results for complex structures and multi-scale data.

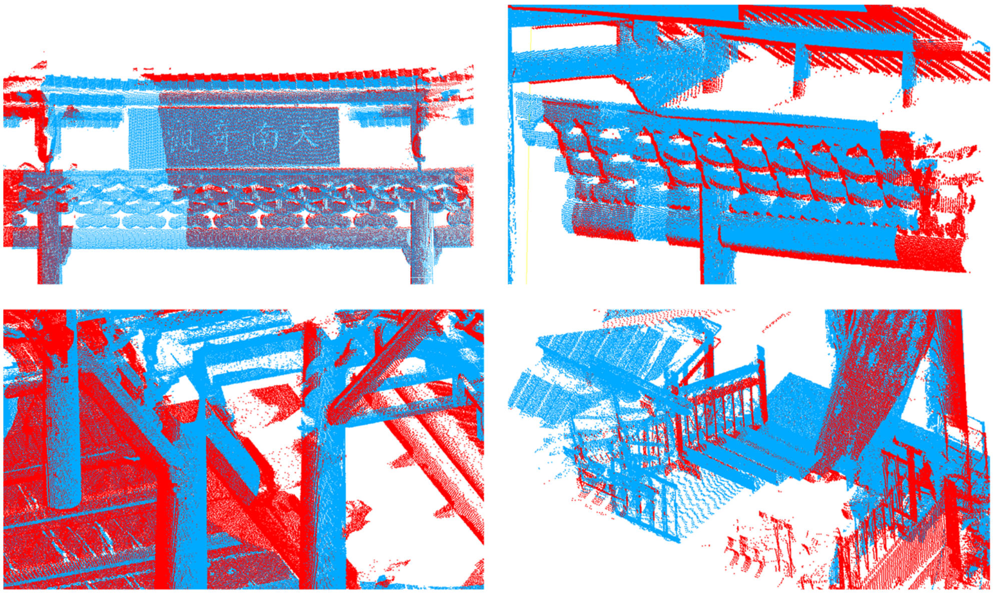


Figure 22. The upper right image shows the nameplate(southern wonder) of the building. The upper left image depicts decorative beams. The lower right image displays beams and pillars within the building. The lower left image shows the junction at the staircase. In this figure, red and blue represent the Zhenwu Temple scanned from different angles.

4. Discussion

Our experimental observations distinctly illustrate the superiority of point-to-point correspondence-based registration methods compared with others. The approach that employs a combination of rigidity constraints and distance of salient point constraints yields a more accurate set of correspondences when contrasted to other geometric constraints. The key advantage of integrating rigidity and DSP constraints is that DSP takes into account local geometric details, effectively resolving ambiguities introduced by normal constraints and rigidity constraints.

When it comes to generating transformation matrices and selecting the best transformation, through experiments, it was shown that the proposed method for creating transformation matrices, based on pairs of points from both the source and target point clouds, outperforms conventional techniques such as RANSAC and GC-SAC. Because of the high-quality correspondences, the proposed method achieves improved efficiency and accuracy in registration results compared to other estimator methods. Additionally, the proposed evaluation method, which combines inlier points and overlap ratio, provides a comprehensive assessment of both local and global qualities of the registration results. This innovative evaluation approach outshines other methods and contributes to a more comprehensive understanding of registration outcomes. Once the registration process was set, we compared our proposed method with three existing feature-based registration methods. The results indicate that our method also outperformed similar registration methods in terms of accuracy. The experiment also proves the robustness of the proposed coarse registration method in real-world applications, as it performs well in terms of precision and practical effectiveness when tested with different types, overlap ratios, and point cloud densities of data.

However, it is important to acknowledge the limitations inherent in our experimental approach. It should be noted that the results, those concerning different estimators, and evaluation techniques, were derived from the Bunny model. To ascertain the method's efficacy across varied datasets, future research should encompass more extensive experiments. Furthermore, in the comparative experiments involving recent methods, we encountered several challenges due to time constraints. Specifically, we encountered the following issues:

(1) Due to our incomplete understanding of the underlying principles of the other three methods, achieving optimal registration results was challenging.
(2) The selection of comparable methods for our study was constrained by a limited pool of options. Additionally, some of the chosen methods may not accurately represent the latest advancements in registration techniques. This aspect limited the breadth and accuracy of our comparative analysis.
(3) By not integrating other established point cloud registration evaluation metrics, the comprehensiveness of our results was compromised, and as a result, the overall persuasiveness of our findings was diminished.

5. Conclusions

In this paper, we propose a coarse registration method based on local geometric feature constraints, combined with a comprehensive evaluation of inliers and overlap ratio. The main steps of this method include correspondences filtering, transformation matrix computation, and evaluation of matrix. First, we combine the constraints of the salient points' distance and rigidity to select high-quality correspondences. Then, based on the centroids of the correspondences and their reference frames, we compute the transformation matrix for each correspondence. Finally, using the evaluation metrics of inliers and overlap ratio, we select the best registration matrix. We compare different descriptors and feature point detection methods to choose the one with the highest registration accuracy as our experimental approach. Additionally, we compare the effects of different geometric constraints on the experimental results and demonstrate that the constraints of salient points' distance and rigidity yield better results. By comparing single evaluation criteria, we show that the overall registration results are improved when both evaluation metrics are considered. Finally, we test our registration method on different types of datasets to demonstrate its robustness and accuracy. In future work, the repetitive evaluation process still significantly consumes time. There is still room for us to optimize the code in order to improve computational efficiency. Additionally, the current method only considers the geometric aspects (XYZ) of the point cloud data. We aim to incorporate additional parameters, such as RGB and intensity, in the subsequent registration experiments to enhance the coarse registration process. These parameters can provide valuable information and improve the accuracy of the coarse registration by considering not only geometric features, but also color and intensity characteristics.

Author Contributions: Supervision, C.K., Z.L., S.Z. (Sai Zhang), S.Z. (Siyao Zhang) and S.W.; writing—original draft, C.G.; writing—review and editing, C.K. All authors have read and agreed to the published version of the manuscript.

Funding: This study was supported by the National Natural Science Foundation of China (41961063, 42064002).

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank the editors and the reviewers for their valuable suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

RMSE	Root Mean Square Error
ICP	Iterative Closest Point
GC-SAC	compatibility-guided sampling consensu
RoPS	Rotation Projection Statistics
3DHV	3D Hough Voting
PCV	Progressive Consistency Voting
ISS	intrinsic shape signatures ISS
3DSC	3D shape context descriptor
GROR	graph reliability outlier removal
RANSAC	Random Sample Consensus
OASC	Optimized Sample Consensus
H3D	Harris 3D
SI	Spin Image
LRF	Local Reference Frame
FPFH	Point Feature Histogram
SHOT	Signature of Histograms of Orientations
NNSR	Nearest Neighbor Similarity Ratio
PCRMLP	point cloud registration with multilayer perceptrons
DBSCAN	density-based spatial clustering of applications with noise
TODLI	triple orthogonal localdepth images
DSP	distance of salient point
PCL	Point Cloud Library
MLS	Mobile Laser Scanning
SVD	Singular Value Decomposition

References

1. Wang, Y.; Solomon, J.M. PRNet: Self-Supervised Learning for Partial-to-Partial Registration. *arXiv* **2019**, arXiv:1910.12240.
2. Chen, J.; Wu, X.; Wang, M.Y.; Li, X. 3D Shape Modeling Using a Self-Developed Hand-Held 3D Laser Scanner and an Efficient HT-ICP Point Cloud Registration Algorithm. *Opt. Laser Technol.* **2013**, *45*, 414–423. [CrossRef]
3. Besl, P.J.; McKay, N.D. A Method for Registration of 3-D Shapes. *IEEE Trans. Pattern Anal. Mach. Intell.* **1992**, *14*, 239–256. [CrossRef]
4. Low, K.L. Linear Least-Squares Optimization for Point-to-Plane ICP Surface Registration. *Chapel Hill* **2004**, *4*, 1–3.
5. Rusinkiewicz, S. A Symmetric Objective Function for ICP. *ACM Trans. Graph. TOG* **2019**, *38*, 85. [CrossRef]
6. Mian, A.S.; Bennamoun; Owens, R.A. Automatic Correspondence for 3d Modeling: An Extensive Review. *Int. J. Shape Model.* **2005**, *11*, 253–291. [CrossRef]
7. Rusu, R.B.; Cousins, S. 3D Is Here: Point Cloud Library (PCL). In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 1–4.
8. Buch, A.G.; Yang, Y.; Krüger, N.; Petersen, H.G. In Search of Inliers: 3D Correspondence by Local and Global Voting. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC, USA, 23–28 June 2014; pp. 2075–2082.
9. Mian, A.S.; Bennamoun, M.; Owens, R. Three-Dimensional Model-Based Object Recognition and Segmentation in Cluttered Scenes. *IEEE Trans. Pattern Anal. Mach. Intell.* **2006**, *28*, 1584–1601. [CrossRef]
10. Lowe, D.G. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
11. Chen, H.; Bhanu, B. 3D Free-Form Object Recognition in Range Images Using Local Surface Patches. In Proceedings of the 17th International Conference on Pattern Recognition, ICPR 2004, Cambridge, UK, 26 August 2004; Volume 3, pp. 136–139.

12. Lakhan, A.; Li, J.; Groenli, T.M.; Sodhro, A.H.; Zardari, N.A.; Imran, A.S.; Thinnukool, O.; Khuwuthyakorn, P. Dynamic Application Partitioning and Task-Scheduling Secure Schemes for Biosensor Healthcare Workload in Mobile Edge Cloud. *Electronics* **2021**, *10*, 2797. [CrossRef]
13. Quan, S.; Yang, J. Compatibility-Guided Sampling Consensus for 3-D Point Cloud Registration. *IEEE Trans. Geosci. Remote Sens.* **2020**, *58*, 7380–7392. [CrossRef]
14. Guo, Y.; Sohel, F.; Bennamoun, M.; Wan, J.; Lu, M. An Accurate and Robust Range Image Registration Algorithm for 3D Object Modeling. *IEEE Trans. Multimed.* **2014**, *16*, 1377–1390. [CrossRef]
15. Yang, J.; Xiao, Y.; Cao, Z.; Yang, W. Ranking 3D Feature Correspondences via Consistency Voting. *Pattern Recognit. Lett.* **2019**, *117*, 1–8. [CrossRef]
16. Sun, L. ICOS: Efficient and Highly Robust Rotation Search and Point Cloud Registration with Correspondences. *arXiv* **2021**, arXiv:2104.14763.
17. Rodolà, E.; Albarelli, A.; Bergamasco, F.; Torsello, A. A Scale Independent Selection Process for 3D Object Recognition in Cluttered Scenes. *Int. J. Comput. Vis.* **2013**, *102*, 129–145. [CrossRef]
18. Tombari, F.; Di Stefano, L. Object Recognition in 3D Scenes with Occlusions and Clutter by Hough Voting. In Proceedings of the 2010 Fourth Pacific-Rim Symposium on Image and Video Technology, Singapore, 14–17 November 2010; IEEE Computer Society: Washington, DC, USA, 2010; pp. 349–355.
19. Sahloul, H.; Shirafuji, S.; Ota, J. An Accurate and Efficient Voting Scheme for a Maximally All-Inlier 3D Correspondence Set. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 2287–2298. [CrossRef] [PubMed]
20. Quan, S.; Yin, K.; Ye, K.; Nan, K. Robust Feature Matching for 3D Point Clouds with Progressive Consistency Voting. *Sensors* **2022**, *22*, 7718. [CrossRef] [PubMed]
21. Xu, G.; Pang, Y.; Bai, Z.; Wang, Y.; Lu, Z. A Fast Point Clouds Registration Algorithm for Laser Scanners. *Appl. Sci.* **2021**, *11*, 3426. [CrossRef]
22. Yan, L.; Wei, P.; Xie, H.; Dai, J.; Wu, H.; Huang, M. A New Outlier Removal Strategy Based on Reliability of Correspondence Graph for Fast Point Cloud Registration. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 7986–8002. [CrossRef]
23. Liang, L.; Pei, H. Affine Iterative Closest Point Algorithm Based on Color Information and Correntropy for Precise Point Set Registration. *Sensors* **2023**, *23*, 6475. [CrossRef]
24. Liu, J.; Xu, Y.; Zhou, L.; Sun, L. PCRMLP: A Two-Stage Network for Point Cloud Registration in Urban Scenes. *Sensors* **2023**, *23*, 5758. [CrossRef]
25. Fischler, M.A.; Bolles, R.C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. In *Readings in Computer Vision*; Fischler, M.A., Firschein, O., Eds.; Morgan Kaufmann: San Francisco, CA, USA, 1987; pp. 726–740, ISBN 978-0-08-051581-6.
26. Quan, S.; Ma, J.; Hu, F.; Fang, B.; Ma, T. Local Voxelized Structure for 3D Binary Feature Representation and Robust Registration of Point Clouds from Low-Cost Sensors. *Inf. Sci.* **2018**, *444*, 153–171. [CrossRef]
27. Yang, J.; Cao, Z.; Zhang, Q. A Fast and Robust Local Descriptor for 3D Point Cloud Registration. *Inf. Sci.* **2016**, *346*, 163–179. [CrossRef]
28. Sipiran, I.; Bustos, B. Harris 3D: A Robust Extension of the Harris Operator for Interest Point Detection on 3D Meshes. *Vis. Comput.* **2011**, *27*, 963–976. [CrossRef]
29. Zhong, Y. Intrinsic Shape Signatures: A Shape Descriptor for 3D Object Recognition. In Proceedings of the 2009 IEEE 12th International Conference on Computer Vision Workshops, Kyoto, Japan, 27 September–4 October 2009; ICCV Workshops: Paris, France, 2009; pp. 689–696.
30. Rusu, R.B.; Blodow, N.; Beetz, M. Fast Point Feature Histograms (FPFH) for 3D Registration. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; pp. 3212–3217.
31. Frome, A.; Huber, D.; Kolluri, R.; Bülow, T.; Malik, J. Recognizing Objects in Range Data Using Regional Point Descriptors. In Proceedings of the Computer Vision—ECCV, Prague, Czech Republic, 11–14 May 2004; Pajdla, T., Matas, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2004; pp. 224–237.
32. Salti, S.; Tombari, F.; Stefano, L.D. SHOT: Unique Signatures of Histograms for Surface and Texture Description. *Comput. Vis. Image Underst.* **2014**, *125*, 251–264. [CrossRef]
33. Johnson, A.E.; Hebert, M. Using Spin Images for Efficient Object Recognition in Cluttered 3D Scenes. *IEEE Trans. Pattern Anal. Mach. Intell.* **1999**, *21*, 433–449. [CrossRef]
34. Yang, J.; Zhang, Q.; Xiao, Y.; Cao, Z. TOLDI: An Effective and Robust Approach for 3D Local Shape Description. *Pattern Recognit.* **2017**, *65*, 175–187. [CrossRef]
35. Yang, J.; Zhang, Q.; Cao, Z. Multi-Attribute Statistics Histograms for Accurate and Robust Pairwise Registration of Range Images. *Neurocomputing* **2017**, *251*, 54–67. [CrossRef]
36. Vallet, B.; Brédif, M.; Serna, A.; Marcotegui, B.; Paparoditis, N. TerraMobilita/IQmulus Urban Point Cloud Analysis Benchmark. *Comput. Graph.* **2015**, *49*, 126–133. [CrossRef]
37. Dias, J.; Simões, P.; Soares, N.; Costa, C.M.; Petry, M.R.; Veiga, G.; Rocha, L.F. Comparison of 3D Sensors for Automating Bolt-Tightening Operations in the Automotive Industry. *Sensors* **2023**, *23*, 4310. [CrossRef] [PubMed]
38. Yang, J.; Xian, K.; Xiao, Y.; Cao, Z. Performance Evaluation of 3D Correspondence Grouping Algorithms. In Proceedings of the 2017 International Conference on 3D Vision (3DV), Qingdao, China, 10–12 October 2017.

39. Zeng, X.L. Research on 3D Point Cloud Registration Algorithm Based on Geometric Features. Master's Thesis, Shandong University, Jinan, China, 2020.
40. Mellado, N.; Aiger, D.; Mitra, N.J. Super 4PCS Fast Global Pointcloud Registration via Smart Indexing. In Proceedings of the Eurographics Symposium on Geometry Processing, Cardiff, UK, 9–11 July 2014; Eurographics Association: Goslar, Germany, 2014; pp. 205–215.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

sensors

MDPI

Article

LiDAR Dynamic Target Detection Based on Multidimensional Features

Aigong Xu, Jiaxin Gao *, Xin Sui, Changqiang Wang and Zhengxu Shi

School of Geomatics, Liaoning Technical University, Fuxin 123000, China; xuaigong@lntu.edu.cn (A.X.); suixin@lntu.edu.cn (X.S.); 471810032@stu.lntu.edu.cn (C.W.); 471920551@stu.lntu.edu.cn (Z.S.)

* Correspondence: 472120751@stu.lntu.edu.cn; Tel.: +86-138-4296-0508

Abstract: To address the limitations of LiDAR dynamic target detection methods, which often require heuristic thresholding, indirect computational assistance, supplementary sensor data, or postdetection, we propose an innovative method based on multidimensional features. Using the differences between the positions and geometric structures of point cloud clusters scanned by the same target in adjacent frame point clouds, the motion states of the point cloud clusters are comprehensively evaluated. To enable the automatic precision pairing of point cloud clusters from adjacent frames of the same target, a double registration algorithm is proposed for point cloud cluster centroids. The iterative closest point (ICP) algorithm is employed for approximate interframe pose estimation during coarse registration. The random sample consensus (RANSAC) and four-parameter transformation algorithms are employed to obtain precise interframe pose relations during fine registration. These processes standardize the coordinate systems of adjacent point clouds and facilitate the association of point cloud clusters from the same target. Based on the paired point cloud cluster, a classification feature system is used to construct the XGBoost decision tree. To enhance the XGBoost training efficiency, a Spearman's rank correlation coefficient-bidirectional search for a dimensionality reduction algorithm is proposed to expedite the optimal classification feature subset construction. After preliminary outcomes are generated by XGBoost, a double Boyer–Moore voting-sliding window algorithm is proposed to refine the final LiDAR dynamic target detection accuracy. To validate the efficacy and efficiency of our method in LiDAR dynamic target detection, an experimental platform is established. Real-world data are collected and pertinent experiments are designed. The experimental results illustrate the soundness of our method. The LiDAR dynamic target correct detection rate is 92.41%, the static target error detection rate is 1.43%, and the detection efficiency is 0.0299 s. Our method exhibits notable advantages over open-source comparative methods, achieving highly efficient and precise LiDAR dynamic target detection.

Keywords: LiDAR dynamic target detection; ICP; RANSAC; XGBoost; Spearman's rank correlation coefficient; feature screening; sliding window; Boyer–Moore voting

1. Introduction

As society enters the era of the mobile Internet, the demand for location services is no longer limited to regional positioning; instead, an all-round positioning service that is not subject to environmental constraints is needed. In outdoor open areas, global navigation satellite systems (GNSSs) can provide mature location services, but these systems have many limitations and deficiencies in complex outdoor environments and indoor environments. Considering these problems, simultaneous localization and mapping (SLAM) has been proposed to promote the gradual maturity of indoor and outdoor integrated high-precision positioning and high-quality spatial data acquisition [1]. Among them, simultaneous localization and mapping based on light detection and ranging (LiDAR SLAM) has the advantages of intuitive mapping, high precision, and the ability to work in all weather conditions [2]. It is widely used in resource exploration, urban planning, agricultural and forestry development, mining inspections, atmospheric detection, and other fields [3–6]. However, practical application scenes often contain dynamic targets. As one of the key steps of LiDAR SLAM, point cloud interframe registration requires the scanning environment to be completely static. Therefore, the point clouds scanned by dynamic targets have different degrees of influence on LiDAR SLAM. If the dynamic target accounts for a large proportion, the SLAM positioning result will have a large error; if the proportion of dynamic targets is relatively small, the dynamic target will interfere less with the accuracy of the point cloud registration, but the point cloud scanned during its movement will leave traces in the final point cloud map, decreasing the mapping quality. Dynamic point clouds not only affect the effectiveness of the above positioning and navigation methods, but also cannot be ignored in the geographic information system (GIS) and remote sensing (RS) fields. If the LiDAR point cloud used to construct a geospatial database or generate a digital elevation model (DEM) contains many dynamic point clouds, the point cloud splicing accuracy and the subsequent application of the database and model will be impacted. When LiDAR point clouds are used for agricultural and forestry census and geological monitoring, there will be a certain degree of error in the corresponding results, because dynamic targets rarely have periodic reproducibility. When LiDAR point clouds are used for building modelling, urban planning, and other tasks, the use of point clouds scanned by dynamic targets does not have research significance; therefore, removing these clouds in advance can effectively improve the efficiency of data processing. In summary, efficiently and accurately detecting and eliminating LiDAR dynamic point clouds are issues of widespread concern in various fields; these tasks are the focus of this paper.

To address the problem of LiDAR dynamic target detection, commonly used methods can be divided into three main categories: segmentation-based methods, visibility-based methods, and voxel-based methods. Segmentation-based methods can be further divided into methods based on traditional clustering segmentation and methods based on learning segmentation. Methods based on traditional clustering segmentation generally use region growing [7], fast point feature histograms (FPFHs) [8], and other algorithms for point cloud clustering segmentation, and then, through point cloud direct registration [9], with the help of other sensors [10], multisource information [11,12], and other methods to unify the adjacent frame point cloud coordinate system, compare the indicators used to judge the motion state of the point cloud cluster and the corresponding threshold to detect dynamic targets. These methods ensure a low data processing complexity, but typically use a single index as the benchmark to judge the motion state of the target, and different thresholds must be set for the corresponding indices in different scenes to obtain effective detection results. Methods based on learning segmentation generally use 3D−MiniNet, RangeNet++, and other networks to directly semantically segment LiDAR point clouds [13,14] or project point clouds into two-dimensional images for indirect semantic segmentation [15–17] and use the obtained semantic labels to detect dynamic targets. These methods are convenient to use and have a wide range of applications. However, constructing a suitable training set, reducing the workload of the training set construction, and improving the training efficiency and accuracy are still key challenges when using these methods. Visibility-based methods are generally based on the physical premise that 'light propagates along a straight line'. The 3D LiDAR point cloud is projected into a 2D distance image [18] or depth image [19]. The single-frame point cloud image is aligned to the corresponding position in the overall point cloud image through perspective coordinate transformation, and the residual image is generated by the pixel difference for dynamic target detection [20]. This kind of method does not require pretraining and is not restricted by the category or number of dynamic targets. However, processing each frame point cloud requires a perspective coordinate transformation, so this kind of method relies on high-precision pose transformation parameters. This method is applicable only to sparse LiDAR point clouds; otherwise, the data processing efficiency is extremely low. In addition, at present, this kind of method still faces two key problems: laser beam physical characteristic interference and static point invisibility. The former is caused by the interference of the point cloud projection effect caused by special LiDAR laser beam structures, such as parallelism and occlusion. The latter problem is caused by the presence of dynamic targets that maintain the same frequency movement as LiDAR during data acquisition. The same frequency target continuously blocks the point cloud behind it; as a result, detecting it effectively through the residual image is impossible. Voxel-based methods can be further divided into probability statistics methods and descriptor comparison methods. Probability statistics methods [21] are also based on the physical premise that 'light propagates along a straight line'. The 3D environment is projected onto the 2D plane and divided into several small grids. As the LiDAR point cloud to be detected accumulates frame by frame, the ray casting-based (RC) algorithm is used to identify the situation where each grid is hit and crossed, and the probability that the grid contains dynamic targets is calculated to screen the dynamic grid. This kind of method achieves the batch detection of the point cloud motion state with the grid as the smallest unit. However, when given a large incident angle of a laser beam or occlusion, the detection results easily become abnormal. Furthermore, detecting each frame point cloud requires traversing all the grids passed by each laser beam, which consumes a large amount of computing resources. The descriptor comparison method requires constructing a global point cloud map or a local point cloud map in advance during the data acquisition process to form a prior reference basis [22,23]. The prior point cloud map and the point cloud to be detected are rasterized. Whether the descriptors of the paired grids differ is determined, and the nonground points in significantly different grids are regarded as dynamic point clouds. This kind of method can effectively detect dynamic targets in various states, but requires a prior map as a reference benchmark; as a result, it is generally used for postdetection, which limits its application. In addition, the grid size directly impacts the detection effect of voxel-based methods. If the grid is too large, part of the static point cloud will be removed, and if the grid is too small, the data processing efficiency will be significantly reduced. In summary, each method has advantages and limitations. It is necessary to study in depth a LiDAR dynamic target detection method with universal applicability that accounts for both detection accuracy and data processing efficiency.

To reduce the influence of heuristic thresholds and auxiliary processes such as point cloud projection and grid segmentation on the final LiDAR dynamic target detection results, based on the point cloud clustering results and the unified adjacent frame point cloud coordinate system, the multidimensional position and geometric structure differences between the paired point cloud clusters are comprehensively considered in this paper. By extracting the characteristics of the point cloud cluster and using a machine learning method to detect the motion state of each point cloud cluster in each frame point cloud, high-precision and high-efficiency LiDAR dynamic target detection is achieved. Among many machine learning algorithms, XGBoost [24] has the advantages of flexibility, accuracy, and efficiency and has been optimized by relevant experts and scholars from the perspectives of data processing, multilabel classification, and hyperparameter tuning [25–28]. Therefore, this algorithm is widely used to address various classification and regression problems [29–31]. However, it is rarely used to detect LiDAR dynamic targets. Therefore, it is taken as the core algorithm of LiDAR target motion state detection in this paper. Compared with other learning-based LiDAR dynamic target detection methods, our method does not require a complex training network structure, training the XGBoost decision tree by reasonably constructing an optimal classification feature subset to detect LiDAR dynamic targets. The proposed method has universal applicability to different environments. When the optimal classification feature subset is constructed, effective feature screening and dimensionality reduction are conducted, which ensures the classification accuracy, compresses the optimal feature subset dimension, and considers the model training efficiency. In addition, the classification label acquisition and classification feature quantification of the training dataset adopt a fully automatic mode, which avoids introducing human error.

The main contributions of this paper are as follows.

- Based on the clustering results of LiDAR point cloud clusters, the geometric center point set of the point cloud clusters of each frame point cloud is taken as the research object, and a double registration algorithm suitable for sparse point clouds is proposed. In the coarse registration stage, the iterative closest point (ICP) algorithm is used to obtain the rough pose relationship between the geometric center point sets of adjacent frame point cloud clusters. In the fine registration stage, the random sample consensus (RANSAC) algorithm and the four-parameter coordinate transformation algorithm are used to calculate a more accurate pose relationship.
- The above pose relationship is used to unify the coordinate datum between the geometric center point sets of adjacent frame point cloud clusters; then, the matching results of the point cloud clusters scanned by the same target are obtained, and the multidimensional position and geometric structure differences are calculated to construct the point cloud cluster classification feature system. Since point cloud cluster features are generally quantitative features, the number of feature splittings (weight) is taken as an indicator of the importance of the features. Moreover, based on Spearman's rank correlation coefficient (SCC), the optimal classification feature subset is constructed by bidirectional feature screening and dimensionality reduction to facilitate both accurate detection and training efficiency for XGBoost.
- Considering that machine learning algorithms have certain mechanical properties and cannot detect dynamic targets with special states, a double Boyer–Moore voting-sliding window scheme based on the sliding window (SW) strategy and the Boyer–Moore voting (BMV) strategy is designed to achieve the secondary correction of the preliminary detection results of XGBoost, thereby improving the accuracy of the final LiDAR dynamic target detection.
- In the corresponding experiments, the effectiveness of the proposed main algorithms is reasonably verified, and it is successfully verified that our method can effectively detect the motion state based on the multidimensional features of adjacent frame paired point cloud clusters, so as to achieve accurate and efficient LiDAR dynamic target detection.

The remainder of this paper is organized as follows. In Section 2.1, the framework of the entire method is outlined. In Section 2.2, a double registration algorithm is proposed to ensure that the point cloud clusters scanned by the same target in two adjacent frames are accurately matched. The extraction and quantification processes of the corresponding classification feature system and the training set classification label when using XGBoost for LiDAR dynamic target detection are described in Section 2.3. In Section 2.4, XGBoost is improved from the perspectives of both model detection accuracy and training efficiency, and a double Boyer–Moore voting-sliding window is designed to correct the preliminary detection results. The detailed experimental setup and discussion are reported and analyzed in Section 3. Finally, in Section 4, the work of this paper is summarized, and the advantages and disadvantages of our method are discussed and delineated.

2. Methodology

In this paper, we use the superscript T to denote the transpose of a vector or matrix, with lowercase bold symbols (e.g., $\boldsymbol{\eta}$) denoting vectors and uppercase bold symbols (e.g., $\boldsymbol{T}$) denoting matrices and collections. For any vector $\boldsymbol{\eta}$, $\|\boldsymbol{\eta}\|$ denotes its Euclidean norm.

2.1. Method Overview

The proposed LiDAR dynamic target detection method based on multidimensional features is shown in Figure 1. The main framework consists of three parts: LiDAR point cloud processing, the construction of a classification feature system and quantification of classification labels, and LiDAR dynamic target detection based on improved XGBoost.

The LiDAR point cloud processing procedure is shown in the first part of Figure 1. Point cloud preprocessing and point cloud cluster segmentation are conducted, and a double registration algorithm suitable for sparse point clouds is proposed to obtain accurate adjacent frame point cloud cluster matching results.

The construction of the classification feature system and the quantification of the classification labels are shown in the second part of Figure 1. The extraction and quantification of differences between the multidimensional positions and geometric structures of the paired point cloud clusters are determined, and a quantification method for classification labels is used to construct the model training dataset.

The LiDAR dynamic target detection process based on improved XGBoost is shown in the third part of Figure 1. In this process, based on traditional XGBoost, weight is used as an indicator of the importance of the features, and a strategy for efficiently constructing an optimal classification feature subset is proposed that considers the training efficiency and detection accuracy of the model. In addition, a secondary correction strategy for the preliminary detection results of XGBoost is proposed to improve the final LiDAR dynamic target detection accuracy.

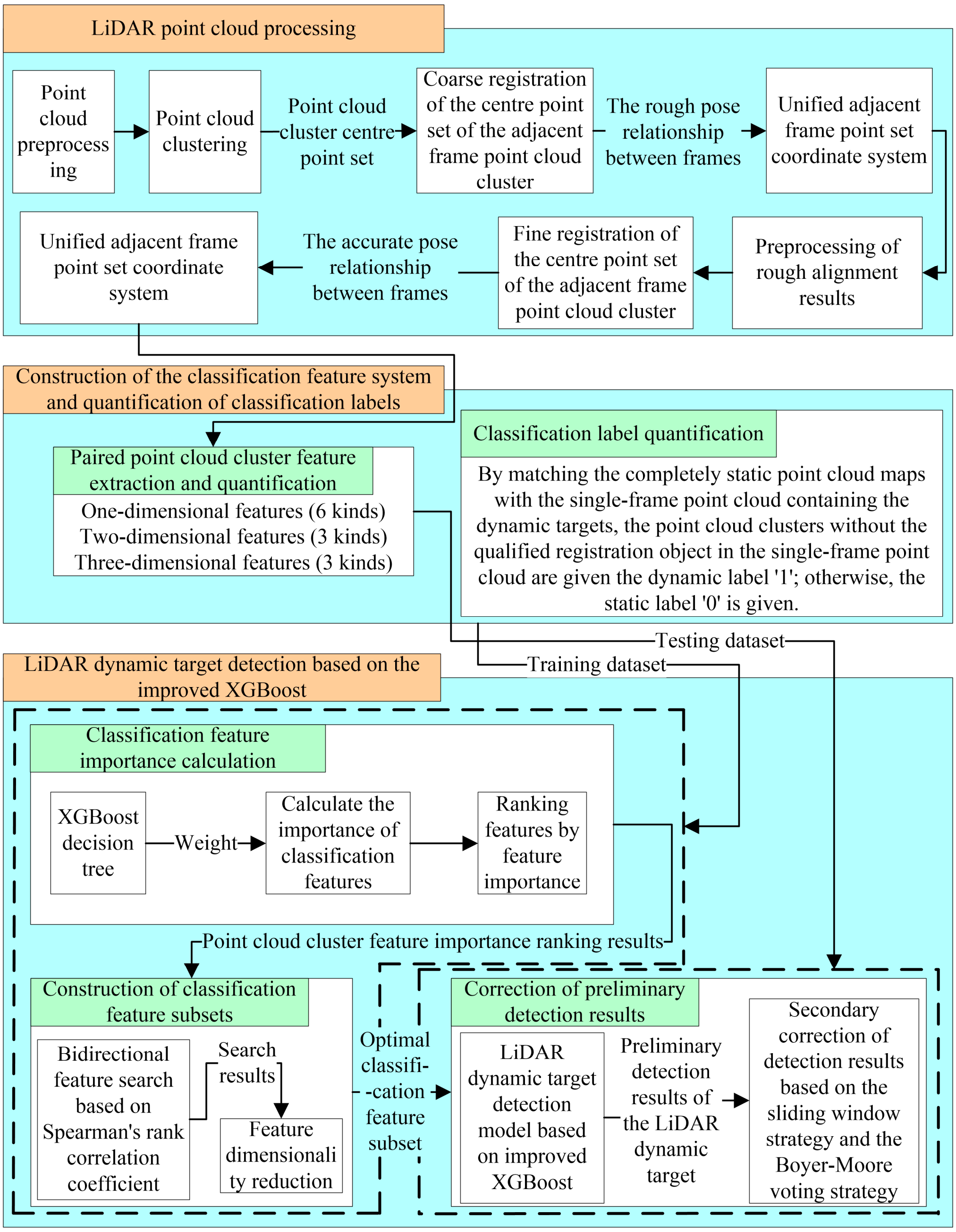

LiDAR point cloud processing
Point cloud preprocessing
Point cloud clustering
Point cloud cluster centre point set
Coarse registration of the centre point set of the adjacent frame point cloud cluster
The rough pose relationship between frames
Unified adjacent frame point set coordinate system
Preprocessing of rough alignment results
Fine registration of the centre point set of the adjacent frame point cloud cluster
The accurate pose relationship between frames
Unified adjacent frame point set coordinate system
Construction of the classification feature system and quantification of classification labels
Paired point cloud cluster feature extraction and quantification
One-dimensional features (6 kinds)
Two-dimensional features (3 kinds)
Three-dimensional features (3 kinds)
Classification label quantification
By matching the completely static point cloud maps with the single-frame point cloud containing the dynamic targets, the point cloud clusters without the qualified registration object in the single-frame point cloud are given the dynamic label '1'; otherwise, the static label '0' is given.
Testing dataset
Training dataset
LiDAR dynamic target detection based on the improved XGBoost
Classification feature importance calculation
XGBoost decision tree
Weight
Calculate the importance of classification features
Ranking features by feature importance
Point cloud cluster feature importance ranking results
Construction of classification feature subsets
Bidirectional feature search based on Spearman's rank correlation coefficient
Search results
Feature dimensionality reduction
Optimal classification feature subset
Correction of preliminary detection results
LiDAR dynamic target detection model based on improved XGBoost
Preliminary detection results of the LiDAR dynamic target
Secondary correction of detection results based on the sliding window strategy and the Boyer-Moore voting strategy

Figure 1. Flowchart of the LiDAR dynamic target detection method based on multidimensional features.

2.2. LiDAR Point Cloud Processing

2.2.1. Construction of the Point Cloud Cluster Center Point Set

LiDAR point clouds exhibit wide coverage, a high measurement accuracy, and dense data, but they also exhibit problems such as equipment system errors [32], point cloud motion distortions [33], and including noise points and discrete points [34]. Therefore, data preprocessing must be performed before LiDAR point clouds can be used for related work. In this paper, an unsupervised LiDAR point cloud optimization algorithm is used to estimate and compensate for system errors in the equipment, and the velocity updating-iterative closest point (V−ICP) method is used to remove point cloud motion distortion [35]. These processing steps greatly improve the quality of the LiDAR point clouds. However, since the purpose of this study is to efficiently detect LiDAR dynamic targets, the high density of the point clouds leads to heavy data processing tasks, so point cloud downsampling is needed. Generally, a point cloud is downsampled by point cloud filtering, removing the noise points and discrete points [36]. Commonly used downsampling methods include Gaussian filtering, direct filtering, voxel filtering, statistical filtering, and bilateral filtering. The effect of direct filtering depends on the heuristic threshold [37]. Gaussian filtering is applicable only to point clouds that follow a normal distribution [38], bilateral filtering is applicable only to ordered point clouds [39], and statistical filtering requires a large amount of calculation [40]. As a result, the voxel filtering method is used in this paper. Constructing a voxel group composed of several three-dimensional hexahedrons ensures that all the LiDAR point clouds of the corresponding frame are completely covered. The number of laser points in each small voxel is counted. The voxels with numbers less than the set threshold are regarded as sparse voxels, and their laser points are regarded as discrete points or noise points and eliminated. For the remaining voxels, the mean value of the point cloud coordinates is calculated separately and used as the voxel center of gravity to replace all the laser points in the current voxel, achieving point cloud downsampling.

Based on the preprocessed LiDAR point cloud, ground segmentation, point cloud cluster clustering, and point cloud cluster geometric center coordinate calculations are conducted using the method provided in Reference [41] to construct the point cloud cluster center point set corresponding to each frame of the LiDAR point cloud. The point cloud cluster center point set corresponding to the frame point cloud is recorded as $\boldsymbol{E}_i$.

2.2.2. Coarse Registration of the Center Point Set of the Adjacent Frame Point Cloud Cluster

The purpose of this study is to judge the motion state of a point cloud cluster based on the multidimensional position and geometric structure difference between the point cloud clusters generated by the same target scanned in the adjacent frame LiDAR point cloud. Therefore, the point cloud cluster matching results generated by the same target scanned from the adjacent frame point cloud cluster set must be accurately obtained. In this paper, this task is equivalent to registering the center point set of the adjacent frame point cloud cluster with high precision, and the point cloud cluster pairing result is obtained based on the registration result of the center point set.

There are often differences between the degrees of position and pose differences of different frame point cloud cluster center point sets that do not pass through the unified coordinate system. As shown in Figure 2, three color point cloud cluster center point sets correspond to three different LiDAR point clouds. Frame A and frame B are adjacent frames, and five frame intervals exist between frame C and frame A. As shown, the more the frames are separated, the more the overall pose differs between the point cloud cluster center point sets. Taking frame A and frame B as examples, if no registration processing is performed and the matching objects of each point in $\boldsymbol{E}_B$ are searched directly in $\boldsymbol{E}_A$, the probability of incorrect matching results is great. In addition, due to the lack of a unified coordinate system, the difference in the position of the point cloud cluster calculated at this time does not have any research significance and cannot be used to judge the motion state of the point cloud cluster. Therefore, a double registration method for the center point set of the point cloud clusters is proposed to accurately solve the pose relationship between $\boldsymbol{E}_A$

and E_B. The double registration method for the point cloud consists of two parts: coarse registration and fine registration. In this section, the coarse registration process is proposed.

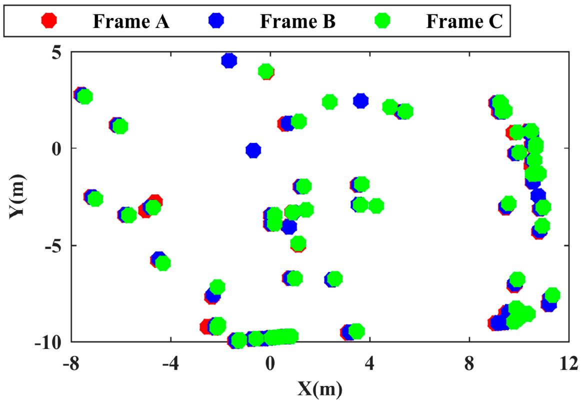


Figure 2. Distribution of point cloud cluster centroid sets for different frames.

Because there is no dramatic change in the pose between frame A and frame B and because the number of points in the center point set of the point cloud cluster is much smaller than the number of points in the entire point cloud, the initial pose matrix of the ICP algorithm [42] is set to a unit matrix, and E_A and E_B are coarsely registered quickly. Based on the rough pose relationship between frames obtained by coarse registration, E_B is converted to the E_A coordinate system. By setting the appropriate search radius, each point in E_B is taken as the research object, and a nearest neighbor search is performed in E_A. The point cloud cluster corresponding to the point without search results is recorded as the 'missing' point cloud cluster. Similarly, using the same search radius, each point in E_A is taken as the research object, and the 'missing' point cloud cluster is reverse-searched in E_B. In addition, after coarse registration and application of the unified coordinate system, the nearest neighbor matching object of each point in the current E_B can be obtained as shown in E_A. To facilitate an intuitive display, in Figure 3, the matching results of some points are shown, and the object shown in the red box is the center point of the known dynamic point cloud cluster.

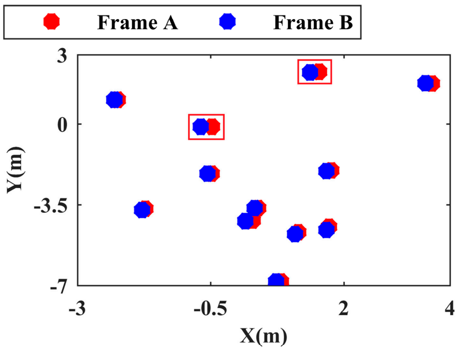


Figure 3. Distribution of the cluster centroids of two adjacent frames in a unified coordinate system after coarse registration.

Because ICP iteratively registers the nearest point and calculates the overall registration error to obtain the optimal solution, it cannot intelligently distinguish the dynamic point cloud cluster center point and the static point cloud cluster center point in the set E; as a result, the registration error of the dynamic point cloud cluster center point impacts the overall registration result. As shown in Figure 3, these problems lead to different degrees of positional differences between the center points of the paired static point cloud clusters after coarse registration. Therefore, the position difference calculated based on the center point of the nearest paired point cloud cluster after coarse registration cannot be used to judge the motion state of the point cloud cluster, and the calculation accuracy of the interframe pose relationship must be further improved.

2.2.3. Fine Registration of the Center Point Set of the Adjacent Frame Point Cloud Cluster

Because the LiDAR laser beam propagates along a straight line, the geometric structure difference largely differs between the point cloud clusters generated by the same target in frame A and frame B due to occlusion or scanning angle changes, which leads to a large degree of positional difference between the center points of the corresponding paired point cloud clusters of some static targets after rough registration. This kind of point cloud cluster is recorded as a pseudodynamic point cloud cluster. To prevent these pseudodynamic point cloud clusters from affecting the fine registration process, the coarse registration results of $\boldsymbol{E}_A$ and $\boldsymbol{E}_B$ are preprocessed, and the center point of the pseudodynamic point cloud cluster is filtered in advance to prevent it from being used as the input for fine registration.

Based on the geometric structure characteristics of each point cloud cluster, the description vector $\boldsymbol{\eta}$ is constructed as follows:

$$\boldsymbol{\eta} = \begin{bmatrix} \varphi_1 & \varphi_2 & \varphi_3 & \varphi_4 \end{bmatrix}^{\mathrm{T}} \tag{1}$$

where φ_1 represents the X-axis span of the point cloud cluster, φ_2 represents the Y-axis span of the point cloud cluster, φ_3 represents the Z-axis span of the point cloud cluster, and φ_4 represents the number of laser points in the point cloud cluster.

The nearest neighbor pairing results of the point cloud clusters after coarse registration are transformed into descriptive vectors in pairs, and the cosine similarity S [43] between vectors is calculated as follows:

$$S = \left(\boldsymbol{\eta}_A^{\mathrm{T}} \cdot \boldsymbol{\eta}_B^k\right) / \left(\|\boldsymbol{\eta}_A\| \left\|\boldsymbol{\eta}_B^k\right\|\right) = \left(\sum_{j=1}^{4}\left(\varphi_A^j \varphi_B^{kj}\right)\right) \Bigg/ \left(\sqrt{\sum_{j=1}^{4}\left(\varphi_A^j\right)^2}\sqrt{\sum_{j=1}^{4}\left(\varphi_B^{kj}\right)^2}\right) \tag{2}$$

where $\boldsymbol{\eta}_B^k$ represents the description vector of the k-th point cloud cluster in $\boldsymbol{E}_B$, $\boldsymbol{\eta}_A$ represents the description vector of the paired point cloud cluster in $\boldsymbol{E}_A$, and j represents the number of elements in the description vector. If S is less than the set threshold, the point cloud cluster being processed in $\boldsymbol{E}_B$ is recorded as pseudodynamic. The above steps are repeated to effectively screen all pseudodynamic point cloud clusters in $\boldsymbol{E}_B$. If there is no geometric structure change between the matching results of the adjacent frame dynamic point cloud clusters, the similarity between the corresponding description vectors must be greater than the set threshold; therefore, the above process cannot effectively filter the dynamic point cloud clusters.

After eliminating the ‘missing’ point cloud clusters and the pseudodynamic point cloud clusters, the center point set of the point cloud cluster is effectively streamlined. Since the pose difference between the point cloud cluster center point sets of frame A and frame B is small after coarse registration and a unified coordinate system is applied, and since several homonymous point pairs are formed by the nearest neighbor pairing of the point cloud cluster center point, four-parameter or seven-parameter coordinate transformation methods can be used for fine registration [44]. In the real scene, there are few targets moving only along the Z-axis direction. The seven-parameter coordinate transformation method requires more calculation than the four-parameter coordinate transformation method. Therefore, in this paper, both $\boldsymbol{E}_A$ and $\boldsymbol{E}_B$ are projected onto the X−Y plane and the four-parameter coordinate transformation method is used to identify the pose relationship. If all the above homonymous point pairs are used to construct the least squares equation to solve the transformation relationship, the influence of dynamic point cloud clusters on the calculation accuracy is also ignored. Therefore, based on the idea of RANSAC [45], two homonymous point pairs are randomly selected to construct the equation and calculate the transformation relationship $\boldsymbol{T}$. The selected point cloud cluster center points are called the initial interior points. $\boldsymbol{T}$ is used to perform coordinate transformation on the remaining points, and the coordinate offset between each point in $\boldsymbol{E}_B$ after transformation and its matching points is calculated. Points below the set threshold are recorded as interior points, and the number of interior points is counted. When the initial interior points are the center points of the point cloud cluster corresponding to the dynamic target, the conversion relationship is calculated, and coordinate transformation is performed. As shown in the red box in Figure 4a, no offset exists between the initial interior points and their matching points; however, a large offset exists between the remaining points and their matching points, and the number of interior points in this case is very small. As shown in Figure 4b, when the initial interior points are the center points of the point cloud cluster corresponding to the static target, the number of interior points is large. A significant offset exists between the center points of the dynamic point cloud cluster and the matching points shown in the red box in the figure, which is normal; a small offset exists between some static point cloud cluster center points and the matching points shown in the green box, which is caused by the occlusion and parallax described above. The process of selecting the initial interior points and counting the number of interior points is iterated until all the combinations of homonymous point pairs are processed, and the process is then stopped. The corresponding $\boldsymbol{T}$ when the number of interior points is the largest is taken as the fine registration result.

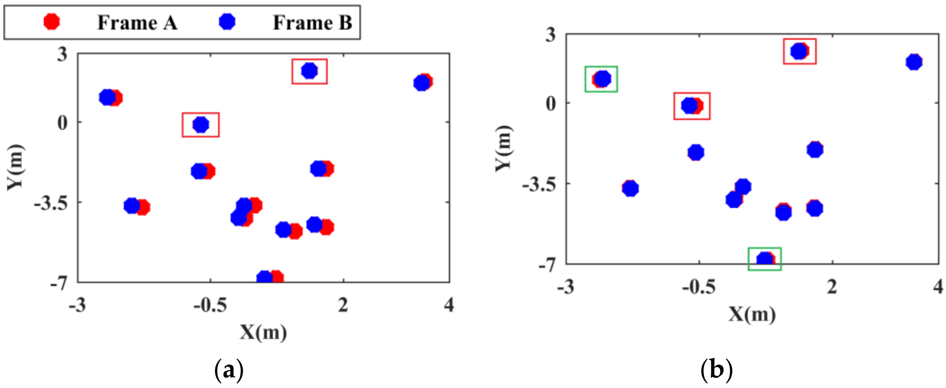


Figure 4. Distribution of the cluster centroids of two adjacent frames in a uniform coordinate system after fine registration: (**a**) dynamic centroids constituting initial interior points; and (**b**) static centroids constituting initial interior points.

2.3. *Construction of the Classification Feature System and Quantification of Classification Labels*

2.3.1. Paired Point Cloud Cluster Feature Extraction and Quantification

Based on the clustering results of the adjacent frame LiDAR point clouds and the matching results of the adjacent frame point cloud clusters after the double registration of the unified coordinate system, 12 kinds of point cloud cluster features that can comprehensively describe the multidimensional position and geometric structure differences between the paired point cloud clusters are extracted and quantified. These features are used to construct the classification feature system. Following the different geometric dimensions of feature extraction, these features are categorized into one-dimensional features, two-dimensional features, and three-dimensional features.

1. A one-dimensional feature refers to a feature extracted based on the difference in position or geometric structure between the paired point cloud clusters in a single dimension of X, Y, and Z. In this paper, six one-dimensional features are extracted, namely, the X-axis displacement D_X, Y-axis displacement D_Y, Z-axis displacement D_Z, X-axis span change degree L_X, Y-axis span change degree L_Y, and Z-axis span change degree L_Z. The methods used to quantify the above six features are as follows.

Taking the point clouds of frame A and frame B after registration as examples, the center point of the k-th point cloud cluster in $\boldsymbol{E}_A$ is denoted as $\left(X_A^k, Y_A^k, Z_A^k\right)$ and the center point of the t-th point cloud cluster paired with the above point cloud cluster in

$\boldsymbol{E}_B$ is denoted as $\left(X_B^t, Y_B^t, Z_B^t\right)$. Using the above variables, D_X, D_Y, and D_Z are calculated as follows:

$$D_X = \left|X_A^k - X_B^t\right| \quad D_Y = \left|Y_A^k - Y_B^t\right| \quad D_Z = \left|Z_A^k - Z_B^t\right| \tag{3}$$

The maximum and minimum values of the X, Y, and Z three-axis coordinates of each laser point in each point cloud cluster are recorded as X_{max}, X_{min}, Y_{max}, Y_{min}, Z_{max}, and Z_{min}, and the three-axis spans H_X, H_Y, and H_Z of the point cloud cluster are calculated as follows:

$$H_X = X_{max} - X_{min} \quad H_Y = Y_{max} - Y_{min} \quad H_Z = Z_{max} - Z_{min} \tag{4}$$

Based on the three-axis span of the point cloud cluster, L_X, L_Y, and L_Z are calculated as follows:

$$L_X = \left(\left|H_{XA}{}^k - H_{XB}{}^t\right|\right)\Big/H_{XA}{}^k \quad L_Y = \left(\left|H_{YA}{}^k - H_{YB}{}^t\right|\right)\Big/H_{YA}{}^k \quad L_Z = \left(\left|H_{ZA}{}^k - H_{ZB}{}^t\right|\right)\Big/H_{ZA}{}^k \tag{5}$$

2. Two-dimensional features are features extracted based on the geometric structure differences between paired point cloud clusters in any two dimensions of X, Y, and Z. In this paper, three two-dimensional features are extracted, namely, the X−Y projection area change degree S_{XY}, the X−Z projection area change degree S_{XZ}, and the Y−Z projection area change degree S_{YZ}. The methods used to quantify the above three features are as follows.

Based on the quantitative results of H_X, H_Y, and H_Z, the projection areas C_{XY}, C_{XZ}, and C_{YZ} of the point cloud clusters in the X−Y plane, X−Z plane, and Y−Z plane are calculated as follows:

$$C_{XY} = H_X H_Y \quad C_{XZ} = H_X H_Z \quad C_{YZ} = H_Y H_Z \tag{6}$$

Based on the plane projection area of the point cloud cluster, S_{XY}, S_{XZ}, and S_{YZ} are calculated as follows:

$$\begin{cases} S_{XY} = \left(\left|C_{XYA}{}^k - C_{XYB}{}^t\right|\right)/C_{XYA}{}^k \\ S_{XZ} = \left(\left|C_{XZA}{}^k - C_{XZB}{}^t\right|\right)/C_{XZA}{}^k \\ S_{YZ} = \left(\left|C_{YZA}{}^k - C_{YZB}{}^t\right|\right)/C_{YZA}{}^k \end{cases} \tag{7}$$

3. A three-dimensional feature is a feature extracted based on the geometric structure differences between the paired point cloud clusters in the X, Y, and Z dimensions. In this paper, three three-dimensional features are extracted, namely, the degree of change in the volume of the point cloud cluster G, the degree of change in the number of laser points M, and the degree of change in the density of the laser points P. The methods used to quantify the above three features are as follows.

Based on the quantitative results of H_X, H_Y, and H_Z, the volume V of the smallest outsourced parallel regular hexahedron of the point cloud cluster is calculated as follows:

$$V = H_X H_Y H_Z \tag{8}$$

Based on the point cloud cluster outsourcing hexahedral volume, G is calculated as follows:

$$G = \left(\left|V_A^k - V_B^t\right|\right)/V_A^k \tag{9}$$

In the above data processing process, the number of laser points N in each point cloud cluster is directly counted, and M is calculated as follows:

$$M = \left(\left|N_A^k - N_B^t\right|\right)/N_A^k \tag{10}$$

Based on the quantitative results of V and the statistical results of N, the laser point density ρ of the point cloud cluster is calculated as follows:

$$\rho = N/V \tag{11}$$

Based on the quantitative results of ρ, P is calculated as follows:

$$P = \left(\left|\rho_A^k - \rho_B^t\right|\right) \Big/ \rho_A^k \tag{12}$$

By analyzing the data and calculation methods required to quantify these 12 features, the corresponding point cloud cluster classification feature system can be constructed hierarchically for each frame of the LiDAR point cloud by traversing the point cloud cluster set once and using multithreading processing, which ensures the timeliness of data processing.

2.3.2. Classification Label Quantification

To ensure the universal applicability of the proposed LiDAR dynamic target detection model, LiDAR point clouds are collected in real indoor and outdoor scenes. By constructing an a priori point cloud map and using the descriptor comparison method based on the voxel-based method, the motion state classification label is automatically assigned to each point cloud cluster in a single-frame point cloud. The specific process is as follows.

Two rounds of data acquisition are conducted in each real scene. The first round is conducted in the period with no dynamic targets or very few dynamic targets in each scene and is used to construct an a priori point cloud map as a reference base. The second round is conducted during the period with more dynamic targets in each scene and is used to quantify the classification feature system and classification labels.

For each data acquisition scene, based on the first data collected, the LiDAR Odometry and Mapping (LOAM) algorithm [46] is used to construct an a priori point cloud map. Because the secondary collected LiDAR point cloud encompasses many dynamic targets, errors will arise when directly detecting dynamic targets directly using the scan-to-map approach [47] to register a single-frame point cloud and a point cloud map. To obtain a more accurate training set classification label, the descriptor comparison method is used to detect the motion state of each point cloud cluster in each frame of the LiDAR point cloud by dividing the grid and comparing the descriptors based on the registered single-frame point cloud and the prior map. The dynamic point cloud cluster and the ′missing′ point cloud cluster are given a dynamic label of '1', and the remaining point cloud clusters are given a static label of '0'.

2.4. LiDAR Dynamic Target Detection Based on the Improved XGBoost

2.4.1. Construction of Classification Feature Subsets

The model training dataset in this paper comprises the above point cloud cluster classification feature system and classification labels. By constructing the objective function and second-order Taylor expansion, the second-order derivative information is used to train the XGBoost classification decision tree, and the model complexity is optimized as a regularization term to ensure a higher model generalizability [48].

The above process outputs three independent feature importance metrics: weight, gain, and cover. Each index can be used as a benchmark for constructing classification feature subsets during model training, which is used to screen important features to improve the classification effect of XGBoost [49]. Because including enumeration features in the point cloud cluster classification feature system constructed in this paper is difficult, weight is selected as the only feature importance measure index, and the features are arranged from large to small according to the weight index, denoted as sequence $\boldsymbol{F}_{ps}$; the arrangement from small to large is denoted as sequence $\boldsymbol{F}_{io}$.

Because the classification feature system contains more features and the number of samples in the training set based on the independent point cloud cluster is large, under the premise of ensuring the detection accuracy of the model, an optimal feature subset construction strategy based on Spearman's rank correlation coefficient-bidirectional search for dimensionality reduction (SCC−BSDR) is proposed in this paper [50]. This approach is used to improve the training efficiency of the model. There are three traditional feature search strategies [51], namely, forward search, backward search, and bidirectional search. Forward and backward searches easily fall into local optimal solutions. Although the bidirectional search is more reliable, its computational complexity is significantly greater than that of the other search strategies.

In this paper, the optimal feature subset construction is based on the bidirectional feature search strategy. On this basis, Spearman's rank correlation coefficient is used to improve the construction efficiency and a dimensionality reduction strategy is used to simplify the composition of the feature subset. First, the most important feature is obtained from $\boldsymbol{F}_{ps}$ and added to the feature subset to construct a classification decision tree. The current classification accuracy is calculated and used as the initial classification accuracy benchmark. The Spearman's rank correlation coefficient between the other features in $\boldsymbol{F}_{ps}$ and the above feature is calculated. The features with correlation coefficients greater than 0.8 are removed and added to the subset in turn, instead of the above feature, and the classification accuracy is then calculated. The corresponding feature with the highest accuracy is retained in the feature subset and the corresponding classification accuracy is used to update the accuracy benchmark and complete the initialization. Subsequently, the most important feature in the current sequence is obtained from $\boldsymbol{F}_{ps}$, and the features with a correlation greater than 0.8 are screened. The above features are added to the feature subset in turn, and the classification accuracy is calculated. The feature subset with the highest classification accuracy and greater than the current accuracy benchmark is retained. If the classification accuracy is below the current accuracy benchmark, no feature is added and no subsequent feature deletion step is performed. The least important feature is obtained from $\boldsymbol{F}_{io}$ in turn, and the corresponding feature is deleted from the current feature subset. If the classification accuracy decreases, the above operation is withdrawn. Otherwise, the above operation is retained, and the accuracy benchmark is updated. The above process is iterated until the features in $\boldsymbol{F}_{io}$ are traversed. The above process of adding and deleting features to and from the feature subset is repeated until $\boldsymbol{F}_{ps}$ is empty. Finally, the optimal classification feature subset and the corresponding XGBoost detection model are obtained for preliminary detection.

2.4.2. Correction of Preliminary Detection Results

Because the XGBoost detection model screens dynamic targets based on the differences between the multidimensional positions and geometric structures of the paired point cloud clusters in adjacent frame LiDAR point clouds, completely and effectively detecting dynamic targets with intermittent static states and first static and then moving states is impossible. Most LiDAR dynamic target online detection methods exhibit these problems. In addition, the optimal classification feature subset constructed in this paper underwent a feature screening step and dimensionality reduction step. Each feature is irreplaceable in the classification process. When the quantitative result of a feature is inaccurate, the XGBoost detection model may yield erroneous detection results. Aiming to address these two problems, a double Boyer–Moore voting-sliding window (DBMV−SW) based on the SW strategy [52] and the BMV strategy is designed to achieve the secondary correction of the preliminary XGBoost test results.

This paper holds that, in an actual situation, a change in motion state in all targets must be a gradual process. A target will not suddenly appear dynamic at a certain moment when its motion state is static, nor will a target suddenly appear static at a certain moment when its motion state is dynamic. In addition, the interframe position change of a point cloud cluster scanned by a dynamic target with an intermittent static state is also a gradual change process during the change in motion state. Therefore, aiming to identify and correct the anomaly detection from the preliminary detection results of XGBoost, the SW strategy is adopted, the point cloud of the i-th frame is taken as the research object, and the five consecutive historical point clouds of frame $i-2, i-3, i-4, i-5, i-6$ are taken as the sliding window range in this paper. The double registration method of the point cloud is used to process the geometric center point set of the point cloud clusters of the i-th frame and each frame point cloud in the window, and the motion state of the five nearest paired point cloud clusters of each point cloud cluster in the window of the i-th frame is obtained. The BMV strategy is used for the first vote for the above motion state corresponding to each point cloud cluster. If the preliminary detection result of adjacent frames is static but the voting result is dynamic, the preliminary detection result of adjacent frames is changed to dynamic. If both the initial detection result of the adjacent frame and the voting result are static, but the motion state of the nearest point cloud cluster with any two consecutive frames or more than two frames in the window is dynamic, the initial detection result of the adjacent frame is changed to dynamic. If the initial detection result of the adjacent frame is dynamic but the voting result is static, this frame must be judged in depth. The first two cases effectively detect dynamic targets with intermittent static states, and the third case effectively detects dynamic targets with static states and then moving states. The preliminary detection results of adjacent frames do not need to be modified in any case besides the above three cases.

Considering the third situation, which needs to be evaluated in depth, the X-axis displacement D_X, Y-axis displacement D_Y, and Z-axis displacement D_Z are taken as dependent variables, and the arrangement number of each frame point cloud in the window is taken as an independent variable to fit the corresponding change trend. Because of the small amount of data and to ensure the efficiency of the data processing, the first-order linear function is used to fit the change trend of the above characteristics. Whether the current adjacent frames D_X, D_Y, and D_Z conform to the corresponding change trend is verified as follows:

$$\Delta C^k = f(\psi) - C^k, \psi = 6 \tag{13}$$

In the formula, ΔC^k and C^k represent the fitting error and quantization result of the k-th feature, respectively, and $f(\psi)$ represents the corresponding fitting function. Because the capacity of the window is 5, when judging whether the features of the current adjacent frame conform to the corresponding feature change trend, the independent variable ψ of the fitting function is 6. When $\Delta C^k > \sigma^k$, the detection result is not consistent with the change trend, and the '0' fitting label is given; otherwise, the '1' fitting label is given. σ represents the standard deviation of the corresponding feature in the window. In addition, the corresponding slope label is also given using the slope of the fitting function corresponding to each feature. If the absolute value of the slope is greater than 1, it is labelled '1', and if the absolute value is less than 1 and greater than 0, it is labelled '0'.

The second vote is based on the fitting label and slope label corresponding to the three features. When a fitting label of '1' corresponds to any feature, if the slope label '1' corresponding to the above features is the majority label, the preliminary detection result of the adjacent frame does not need to be changed. In addition, any label combination must change the preliminary detection result of the adjacent frame to be static. The center point global coordinates of all the point cloud clusters that do not require changing the initial detection results of the adjacent frames during the detection process are recorded to form a set $\boldsymbol{W}$. All the adjacent retrieval point cloud clusters belonging to $\boldsymbol{W}$ in the global coordinate system are regarded as dynamic point cloud clusters.

3. Experiments and Analysis

3.1. Source of Experimental Data

To verify whether the SCC−BSDR effectively considers the classification accuracy and model training efficiency of XGBoost, six datasets containing only quantitative features are randomly selected from the UCI machine learning database [53] (http://archive.ics.uci.

edu/ accessed on 9 August 2023) as test data, including the wine, wireless indoor locating, iris, banknote authentication, abalone, and EEG eye state datasets.

To prevent overfitting by the training results of the LiDAR dynamic target detection model, the number of dynamic target samples and the number of static target samples contained in the training set should not differ by more than 200%. In this paper, an experimental platform is built based on LiDAR and Inertial Measurement Unit (IMU). LiDAR is used to collect the data for constructing the training set of the model in this paper, and IMU is used to collect the data needed for the comparison method. Using the above experimental platform, experimental data are collected from five real indoor and outdoor scenes, including an indoor warehouse scene with a lower people flow, an indoor teaching building scene with a higher people flow, an outdoor playground scene with a higher people flow, an outdoor campus trunk road scene with a higher traffic flow, and an indoor underground garage scene with a lower traffic flow. In addition, five common real indoor and outdoor scenes are selected as test scenes to verify the effectiveness of DBMV−SW and the detection effect of our method on LiDAR dynamic targets.

3.2. Experimental Platform and Experimental Scenes Overview

The experimental platform is shown in Figure 5a. LiDAR uses Velodyne's VLP−16, which has a horizontal scanning field of view of 360°, a vertical scanning field of view of 30°, and collects data at a frequency of 10 Hz. The IMU uses Ellipse−N from the SBG. The bias stability and repeatability of the gyroscope are 0.1 deg/s, the bias stability and repeatability of the accelerometer are 5 mg and 0.6 mg, respectively, and the data are collected at a frequency of 100 Hz. Test scene 1 is an indoor shopping mall, as shown in Figure 5b, which is distributed with static features such as models, stairs, and billboards. Test scene 2 is an outdoor bustling road section, as shown in Figure 5c. The road surface is flat, and there are shops on both sides. Between the shops and the motor vehicle lanes, there are steps, auxiliary roads, and green belts, and static features such as trash cans, isolation piers, street lamps, and substation boxes are distributed throughout the scene. Test scene 3 is an outdoor square, as shown in Figure 5d. This scene is distributed with static features such as trash cans, benches, and fitness equipment. Test scene 4 is an outdoor playground, as shown in Figure 5e. There are static features such as ball racks, models, a flag raising platform, and trash cans inside the site. Test scene 5 is an underground parking lot, as shown in Figure 5f. Static features such as load-bearing columns, fire hydrants, and debris piles are distributed inside the site. In scene 1, the dynamic targets are mainly pedestrians, and the people flow is high. Due to the indoor scene, the static objects are seriously occluded by dense dynamic targets to verify whether the target motion state detection effect of our method in the indoor environment is robust. The dynamic targets in scene 2 are mainly pedestrians and cars, which also have a high flow. Due to the outdoor environment and the existence of large dynamic targets, the occlusion problems of dynamic targets to static objects and between dynamic targets are more serious in this scene to verify whether the dynamic target detection effect of our method in complex outdoor environments is robust. The dynamic targets in scene 3 are mainly pedestrians. Although the flow of people is high, no large dynamic target exists, and this scene is outdoor and open. Therefore, the occlusion problem between targets can be ignored to verify whether the dynamic target detection effect of our method in an open outdoor environment is robust. The dynamic targets in scene 4 are mainly pedestrians. Similarly, the people flow is high and belongs to the outdoor open scene, but it belongs to the rainy weather. It is used to verify whether the target motion state detection effect of our method in a non-ideal outdoor environment is robust. In scene 5, the dynamic targets are mainly pedestrians and cars with less traffic, which belongs to the weak illumination scene. It is used to verify whether the target motion state detection effect of our method in a non-ideal indoor environment is robust. The plane structure of the above test experimental scene and the data acquisition route of the test set are shown in Figure 6a–e. The ground object categories represented by each symbol in the figure are shown in the legend.

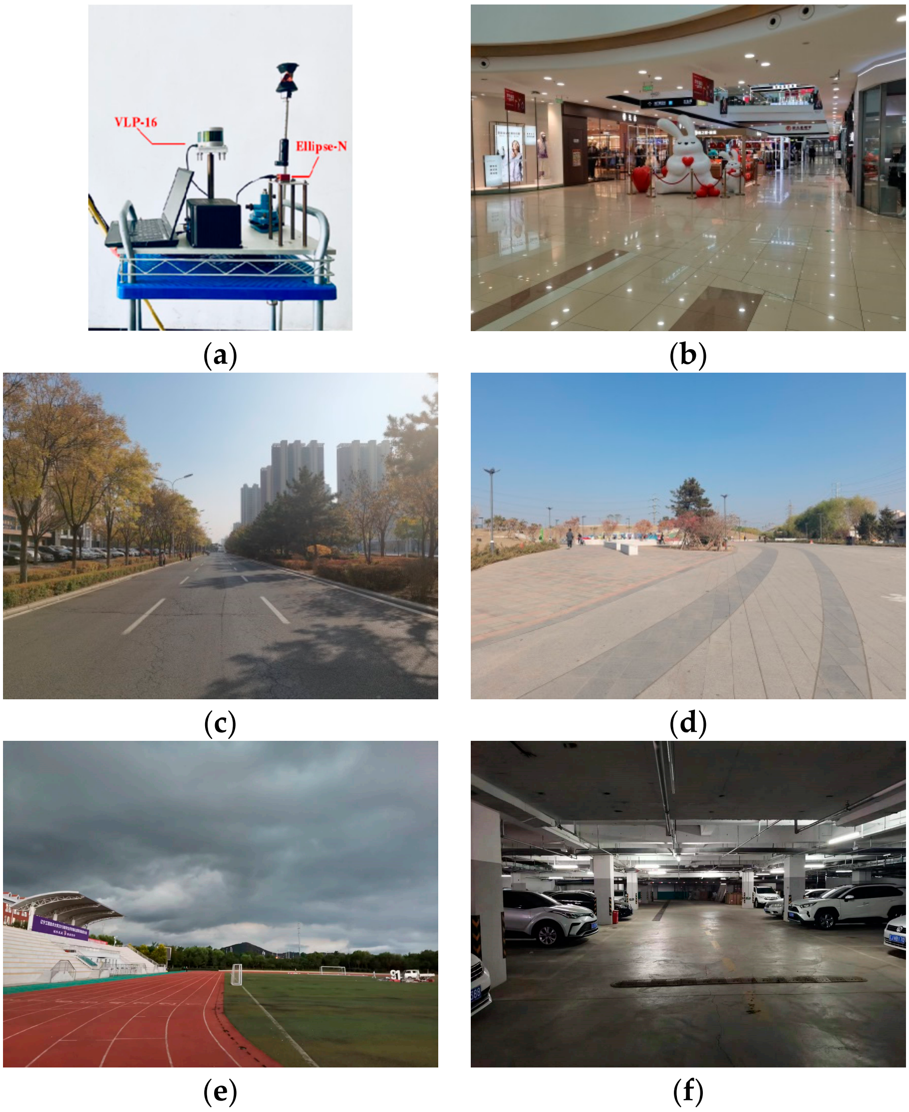


Figure 5. Experimental platform and scenes: (**a**) experimental platform; (**b**) experimental scene 1; (**c**) experimental scene 2; (**d**) experimental scene 3; (**e**) experimental scene 4; and (**f**) experimental scene 5.

3.3. Validation Experiment of the SCC−BSDR and Analysis

To verify whether the SCC−BSDR leads traditional XGBoost to account for both training efficiency and classification accuracy, for each dataset obtained from the UCI machine learning database, 70% of the data are used as the model training set, and the precision, recall, F1-measure, accuracy, and training time (*Time*) of the improved XGBoost algorithm on each test set are calculated. Based on the concept of ablation experiments, four algorithms are designed and compared with the algorithm in this paper (number 5):

- Algorithm 1: Forward search XGBoost algorithm based on weight
- Algorithm 2: Backward search XGBoost algorithm based on weight
- Algorithm 3: Bidirectional search XGBoost algorithm based on weight
- Algorithm 4: Bidirectional search XGBoost algorithm based on weight (including SCC−BS)
- Algorithm 5: Bidirectional search XGBoost algorithm based on weight (including SCC−BSDR) (our algorithm)

Algorithm 3 is compared with algorithm 1–2 to verify the effectiveness of the BS strategy; our algorithm is compared with algorithm 3 to verify the effectiveness of the

SCC−BSDR strategy; and our algorithm is compared with algorithm 4 to verify the effectiveness of the DR strategy. The F1-measure and *Time* corresponding to each algorithm are shown in Table 1. In addition, taking the EEG Eye State and Wine datasets as examples, the effectiveness of our algorithm is intuitively illustrated through radar charts, as shown in Figure 7. The training efficiency index $Efficency$ in the figure is calculated as follows:

$$Efficency = (Time_{max} - Time)/Time_{max} \tag{14}$$

where $Time_{max}$ represents the maximum value of the *Time* index of each algorithm corresponding to the current dataset.

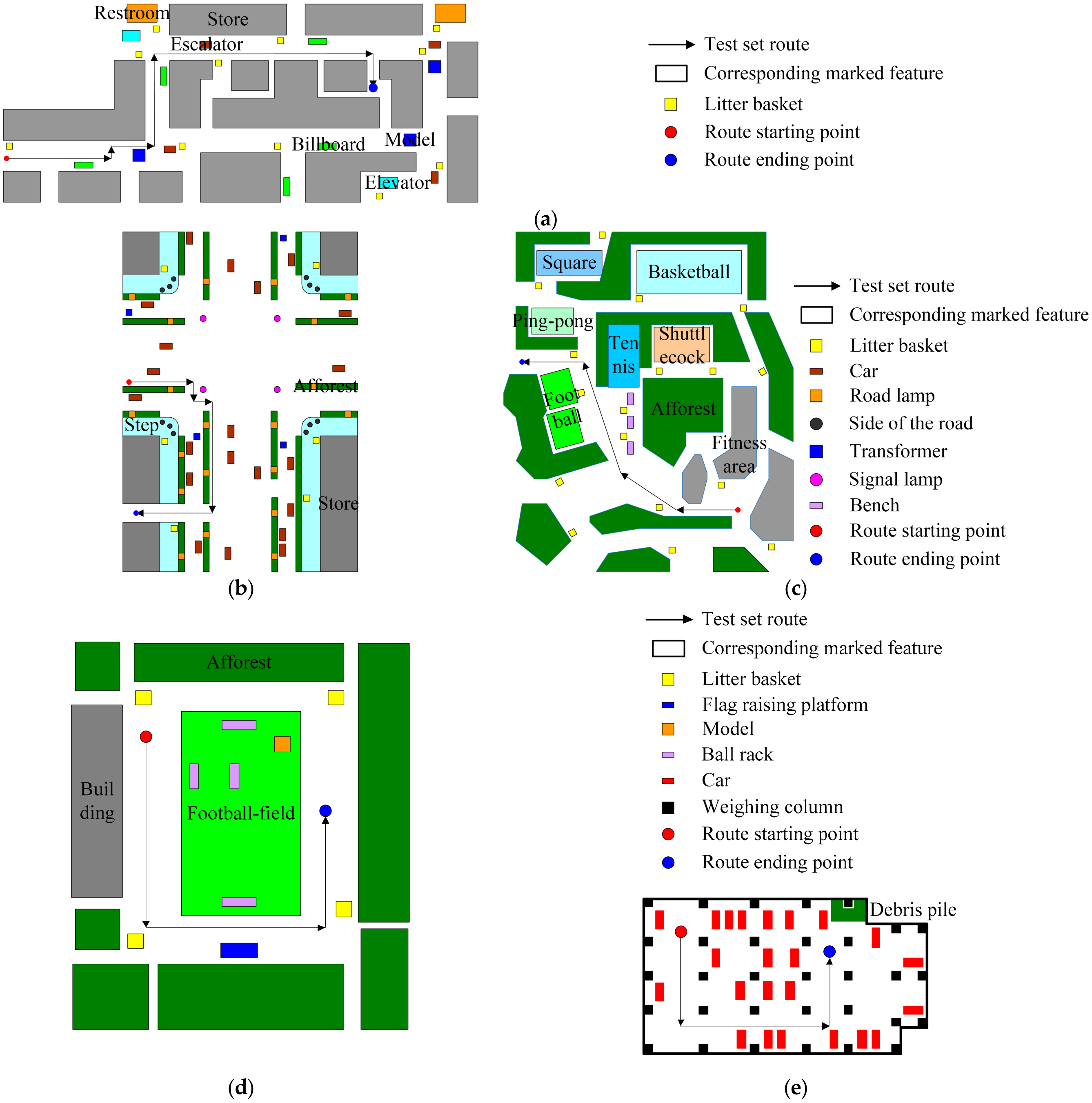
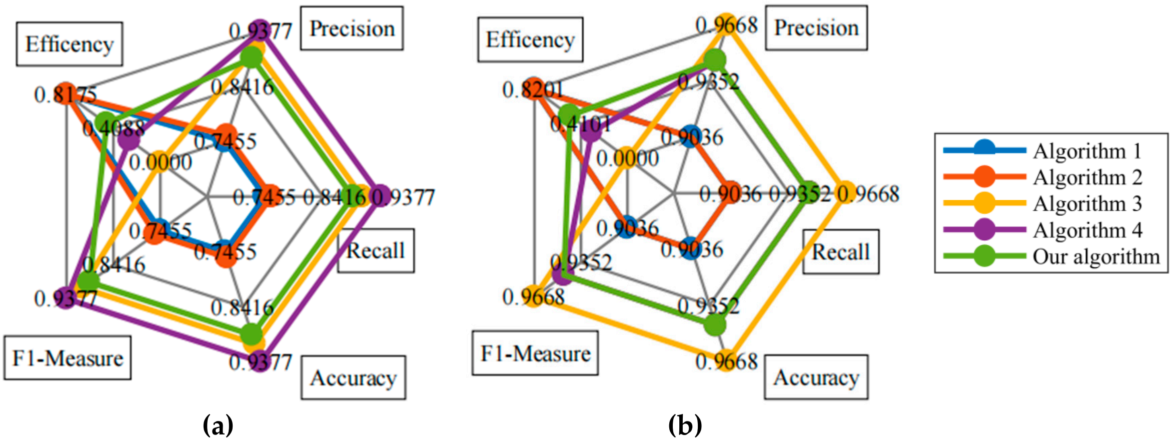


Figure 6. Experimental scene floor plans and platform motion trajectories: (**a**) experimental scene 1; (**b**) experimental scene 2; (**c**) experimental scene 3; (**d**) experimental scene 4; and (**e**) experimental scene 5.

Table 1. Comparison of the classification accuracy and training efficiency of each algorithm.

Dataset	Contrast Ratio	Algorithm 1	Algorithm 2	Algorithm 3	Algorithm 4	Our Algorithm
Wine	F1-measure	0.9036	0.9036	0.9668	0.9467	0.9467
	Time (seconds)	2.6583	2.7273	14.7758	10.1208	7.2954
Wireless Indoor Locating	F1-measure	0.8785	0.8785	0.9667	0.9869	0.9376
	Time (seconds)	3.2112	2.9591	18.5169	13.5056	9.0544
Iris	F1-measure	0.9474	0.8998	0.9474	0.9237	0.9259
	Time (seconds)	2.2724	1.8457	7.8570	5.9028	2.6787
Abalone	F1-measure	0.9265	0.9176	0.9609	0.9627	0.9454
	Time (seconds)	3.2858	3.2993	15.7363	11.8754	7.8578
Banknote Authentication	F1-measure	0.9269	0.9269	0.9562	0.9562	0.9506
	Time (seconds)	3.6781	3.3008	15.8974	12.6205	8.0695
EEG Eye State	F1-measure	0.7455	0.7559	0.9063	0.9377	0.8908
	Time (seconds)	5.9584	5.9702	32.6547	23.9361	17.3668
Mean Value	F1-measure	0.8881	0.8804	0.9507	0.9523	0.9328
	Time (seconds)	3.5107	3.3504	17.5730	12.9935	8.7204



(a) (b)

Figure 7. Comprehensive evaluation radar charts of each algorithm: (**a**) EEG eye state dataset; and (**b**) wine dataset.

Intuitively comparing the areas surrounded by the corresponding closed curves of each algorithm in Figure 7 shows that, on the two datasets, the area corresponding to our algorithm is always the largest, indicating that our algorithm can consider both model classification accuracy and training efficiency for quantitative datasets. The above viewpoint can be further confirmed by the indices shown in Table 1. The F1-measures of Algorithm 1 on the wine, iris, abalone, and banknote authentication datasets are greater than 0.9; the F1-measures of Algorithm 2 are greater than 0.9 only on the wine, abalone, and banknote authentication datasets; and the F1-measures of Algorithm 3, Algorithm 4, and our algorithm are greater than 0.89 and generally greater than 0.9 on each dataset. The training time required by the above three algorithms is greater than that required by algorithm 1 or algorithm 2, but the average training efficiency of our algorithm is 50.38% and 32.89% greater than that of algorithm 3 and algorithm 4, respectively. In addition, algorithm 3 and algorithm 4 achieve similar detection accuracies. The average classification accuracy of our algorithm is 1.88% and 2.05% lower than that of algorithm 3 and algorithm 4, respectively, but the efficiency improvement and accuracy reduction ratios are 26.7979 and 16.0439, respectively. Comprehensive analysis reveals that, although the model with the one-way search strategy achieves a high training efficiency, this model easily falls into the local optimal solution. The detection accuracies of algorithm 3, algorithm 4, and our algorithm are significantly higher than those of the other algorithms because of the bidirectional feature search strategy. Algorithm 4 uses the SCC−BS strategy to accelerate the construction of feature subsets, so it achieves a higher training efficiency than algorithm 3. Our algorithm uses the SCC−BSDR strategy to accelerate the construction of feature subsets and reduce their dimensionality. The optimal classification feature subset required for our algorithm to construct the classification decision tree is simpler than that for algorithm 4. Although the classification accuracy of a small part of the model is sacrificed, the model training efficiency is significantly improved. In summary, the effectiveness of SCC−BSDR is successfully verified by considering both classification accuracy and training efficiency.

3.4. Validation Experiment of the DBMV−SW and Analysis

In this paper, the DBMV−SW algorithm is proposed to correct the preliminary detection results of LiDAR dynamic targets based on the improved XGBoost algorithm. This compensates for the mechanicality of the machine learning algorithm and effectively detects dynamic targets with intermittent static states and first static and then moving states, improving the final detection accuracy. Since scene 2 contains many dynamic targets with the above special states, the data collected in this scene are used as test data and a DBMV−SW validity verification experiment is designed.

The LiDAR dynamic target detection method based on multidimensional features and the detection method excluding DBMV−SW are used to detect the dynamic target of each frame of the LiDAR point cloud in the dataset, and the static point cloud map is used as a reference. The number of static target benchmarks NUM_s, the number of dynamic target benchmarks NUM_d, the number of static target false detections num_s, and the number of dynamic target missed detections num_d of all frame point clouds and each frame point cloud in the dataset are counted. The overall dynamic target correct detection rate η_d and the static target error detection rate η_s of all frame point clouds are calculated by Formula (15), as shown in Table 2. According to the total number of point cloud frames contained in the dataset and the total processing time, the time required to process a frame of point cloud is calculated, which is recorded as the average detection efficiency $time$, and the real-time measurement index η_t is calculated by Formula (15), as shown in Table 2.

$$\eta_s = num_s / NUM_s \quad \eta_d = num_d / NUM_d \quad \eta_t = 1 - (time/0.1) \tag{15}$$

In the formula, 0.1 represents the corresponding inter-frame time interval when the point cloud is collected at the frequency of 10 Hz.

Table 2. The influence of DBMV−SW on detection accuracy and efficiency.

Index Types	Our Method (without DBMV−SW)	Our Method
The total number of dynamic target missed detections	20,852	12,450
The total number of static target false detections	4420	3786
The dynamic target correct detection rate	83.67%	90.25%
The static target error detection rate	2.37%	2.03%
Detection efficiency (seconds)	0.0258	0.0316
Real-time measurement index	74.21%	68.42%

To visually compare the detection effects of the two methods, the dynamic target correct detection rate and the static target error detection rate corresponding to each frame point cloud are calculated by Formula (15), and time-varying sequence diagrams of the detection effects of the two methods are drawn, as shown in Figure 8.

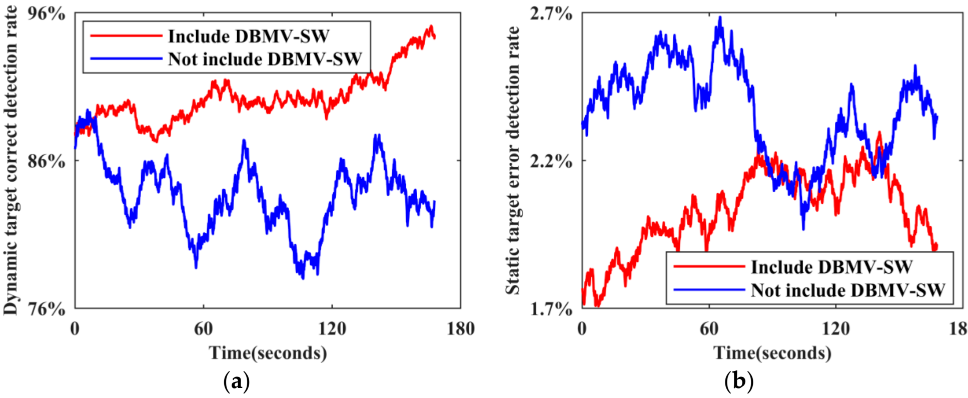


Figure 8. Time-varying sequence diagram of the detection effect: (**a**) dynamic target correct detection rate; and (**b**) static target error detection rate.

As shown in Figure 8, after the secondary correction of the preliminary detection results by DBMV−SW, the dynamic target correct detection rate is generally above 0.85. Although the detection accuracy of individual frames is low, it is higher than the average detection accuracy without correction. In addition, the static target error detection rate is lower than 0.025, which is better than that in the case without correction. Combining this figure with Table 2 for a comprehensive analysis reveals that, after using the DBMV−SW strategy, the number of dynamic target missed detections is 40.29% lower and the number of static target false detections is 14.34% lower. Since DBMV−SW contains steps such as first voting, building tags, and second voting, and each step must be executed in a fixed order, it consumes part of the operation time on the basis of the preliminary detection based on the improved XGBoost, resulting in the average detection efficiency reduction of 22.48% and a real-time measurement index reduction of 7.80%. However, the ratio of the overall detection accuracy improvement to the efficiency reduction is 1.59, indicating that a small part of efficiency can be sacrificed to improve accuracy. The effectiveness of DBMV−SW is thus successfully verified.

3.5. Experiment of LiDAR Dynamic Target Detection Based on Multidimensional Features

Because the LiDAR point cloud in the dataset of this paper is dense, the visibility-based method is not suitable for dynamic target detection. In addition, voxel-based methods are all postprocessing methods. The process of constructing the reference benchmark for the training set employs this kind of method; as a result, it cannot be used as a comparison method. Therefore, two open-source segmentation-based methods, including a traditional clustering segmentation method and a learning segmentation method, are selected as comparison methods. Specifically, the LiDAR dynamic point cloud detection method based on calculating the similarity scores of the corresponding clusters in adjacent frames described in Reference [41] is selected as comparison method 1, and the deep-learning-based LiDAR dynamic point cloud detection method described in Reference [14] is selected as comparison method 2. Based on the data collected in three test scenes, the LiDAR dynamic target detection method based on multidimensional features proposed in this paper is comprehensively evaluated by comparing the detection accuracy and detection efficiency of these methods.

Using the reference benchmark of each scene, the number of static target benchmarks and the number of dynamic target benchmarks of each frame point cloud and all frame point clouds in the corresponding dataset are counted, respectively, as are the number of static target false detections and the number of dynamic target missed detections for each method. Following the calculation method of the dynamic target correct detection rate and the static target error detection rate of each frame point cloud described in Section 3.4, time−varying sequence diagrams of the corresponding detection effects of the three methods in each scene are drawn, as shown in Figures 9–13. The overall dynamic target correct detection rate, static target error detection rate, detection efficiency, real-time measurement index, and mean values of the above indicators of each method on each dataset are calculated, as shown in Table 3. In addition, the dynamic target is regarded as the positive sample, and the static target is regarded as the negative sample. The overall target motion state detection Accuracy, Precision, Recall, F1-Measure, and mean values of the above indicators of each method on each dataset are calculated, as shown in Table 3.

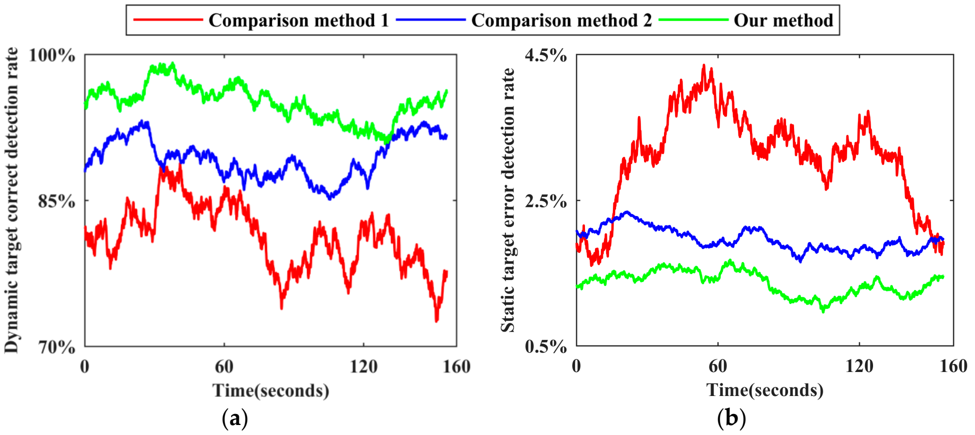


Figure 9. Time-varying sequence diagram of the detection effect of the three methods in scene 1: (**a**) dynamic target correct detection rate; and (**b**) static target error detection rate.

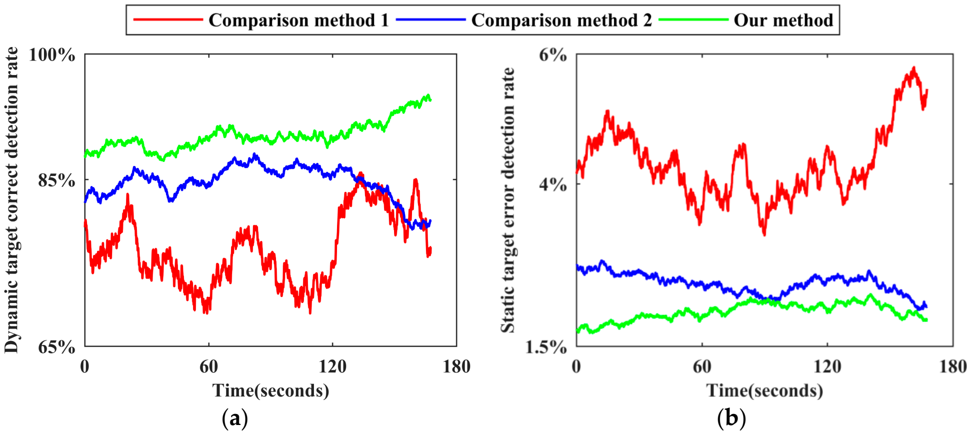


Figure 10. Time-varying sequence diagram of the detection effect of the three methods in scene 2: (**a**) dynamic target correct detection rate; and (**b**) static target error detection rate.

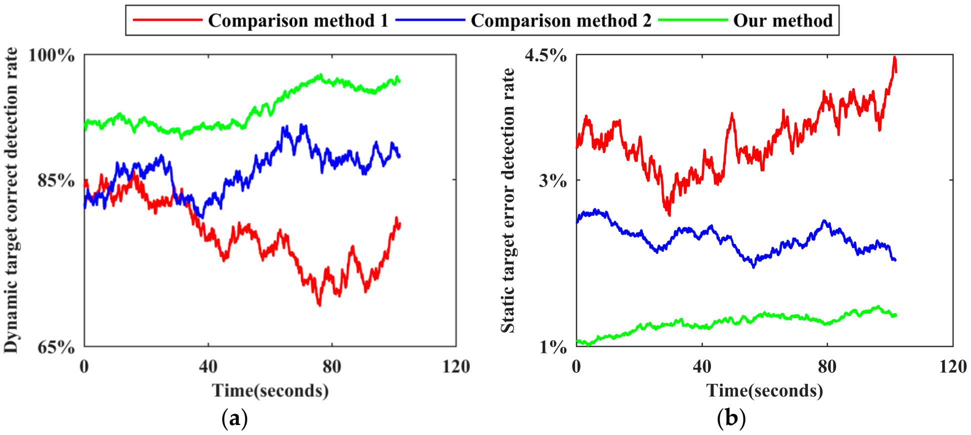


Figure 11. Time-varying sequence diagram of the detection effect of the three methods in scene 3: (**a**) dynamic target correct detection rate; and (**b**) static target error detection rate.

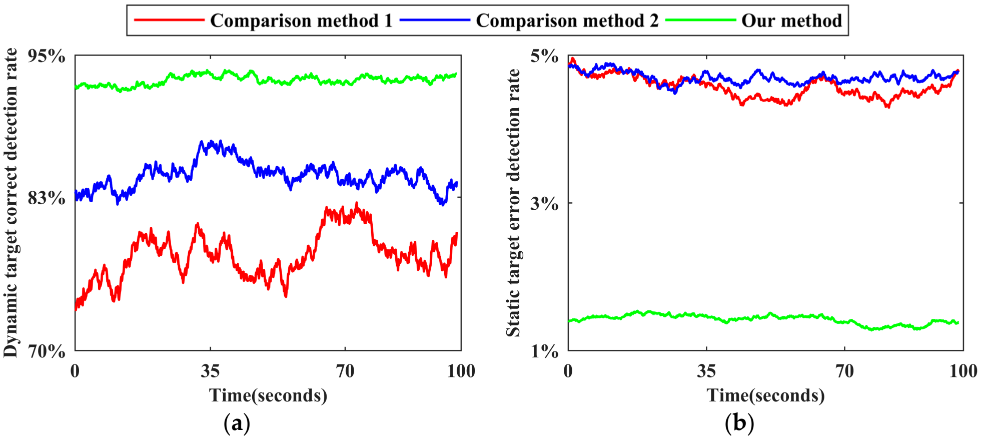


Figure 12. Time-varying sequence diagram of the detection effect of the three methods in scene 4: (**a**) dynamic target correct detection rate; and (**b**) static target error detection rate.

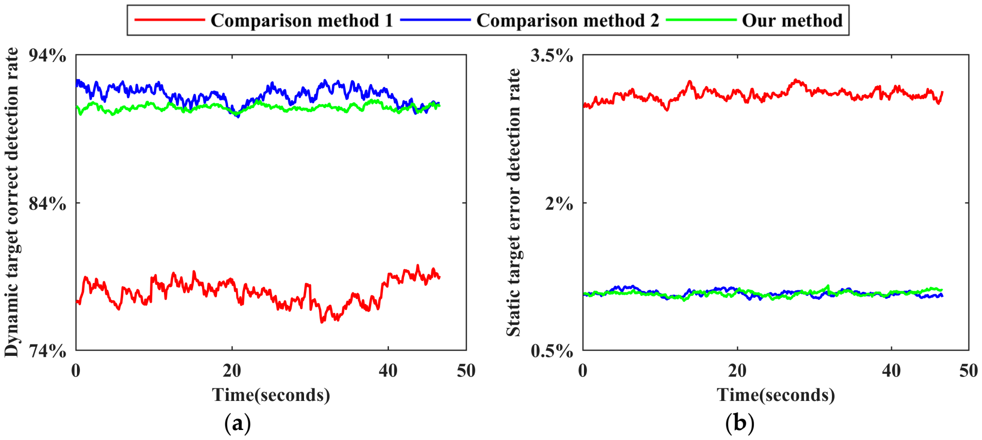


Figure 13. Time-varying sequence diagram of the detection effect of the three methods in scene 5: (**a**) dynamic target correct detection rate; and (**b**) static target error detection rate.

Table 3. Comparison of the detection effect and efficiency of the comparison method and our method.

Index Types	Experimental Scene	Comparison Method 1	Comparison Method 2	Our Method
The total number of dynamic target missed detections	Scene 1	12,286	6972	3230
	Scene 2	29,904	19,600	12,450
	Scene 3	27,150	17,527	8205
	Scene 4	19,320	13,340	6271
	Scene 5	4427	1756	1922
	Mean value	18,617	11,839	6416
The total number of static target false detections	Scene 1	2114	1346	934
	Scene 2	8056	4587	3786
	Scene 3	4000	2640	1475
	Scene 4	4335	4459	1347
	Scene 5	504	176	175
	Mean value	3802	2642	1543
The dynamic target correct detection rate	Scene 1	81.25%	89.36%	95.07%
	Scene 2	76.58%	84.65%	90.25%
	Scene 3	78.36%	86.03%	93.46%
	Scene 4	78.03%	84.83%	92.87%
	Scene 5	77.91%	91.24%	90.41%
	Mean value	78.43%	87.22%	92.41%

Table 3. *Cont.*

Index Types	Experimental Scene	Comparison Method 1	Comparison Method 2	Our Method
The static target error detection rate	Scene 1	3.08%	1.96%	1.36%
	Scene 2	4.32%	2.46%	2.03%
	Scene 3	3.47%	2.29%	1.28%
	Scene 4	4.57%	4.70%	1.42%
	Scene 5	3.09%	1.08%	1.07%
	Mean value	3.71%	2.50%	1.43%
The target motion state detection accuracy	Scene 1	89.27%	93.80%	96.90%
	Scene 2	87.92%	92.30%	94.83%
	Scene 3	87.06%	91.62%	95.98%
	Scene 4	87.06%	90.26%	95.83%
	Scene 5	86.44%	94.69%	94.23%
	Mean value	87.55%	92.53%	95.55%
The target motion state detection precision	Scene 1	96.18%	97.75%	98.52%
	Scene 2	92.39%	95.93%	96.82%
	Scene 3	96.09%	97.61%	98.76%
	Scene 4	94.06%	94.36%	98.38%
	Scene 5	96.87%	99.05%	99.04%
	Mean value	95.12%	96.94%	98.30%
The target motion state detection recall	Scene 1	81.25%	89.36%	95.07%
	Scene 2	76.58%	84.65%	90.25%
	Scene 3	78.36%	86.03%	93.46%
	Scene 4	78.03%	84.83%	92.87%
	Scene 5	77.91%	91.24%	90.41%
	Mean value	78.43%	87.22%	92.41%
The target motion state detection F1-measure	Scene 1	88.09%	93.37%	96.77%
	Scene 2	83.75%	89.94%	93.42%
	Scene 3	86.32%	91.46%	96.04%
	Scene 4	85.30%	89.34%	95.54%
	Scene 5	86.36%	94.98%	94.53%
	Mean value	85.96%	91.82%	95.26%
Detection efficiency (seconds)	Scene 1	0.0386	0.0315	0.0289
	Scene 2	0.0462	0.0357	0.0316
	Scene 3	0.0415	0.0348	0.0302
	Scene 4	0.0427	0.0345	0.0307
	Scene 5	0.0375	0.0309	0.0282
	Mean value	0.0413	0.0335	0.0299
Real-time measurement index	Scene 1	61.39%	68.48%	71.09%
	Scene 2	53.77%	64.26%	68.39%
	Scene 3	58.48%	65.19%	69.83%
	Scene 4	57.29%	65.51%	69.32%
	Scene 5	62.52%	69.13%	71.79%
	Mean value	58.69%	66.51%	70.08%

Comparison method 1 judges the motion state of the target based on the change in the velocity and position of the adjacent frame paired point cloud cluster. For dynamic targets in intermittent static states and those first in static and then dynamic states, this method cannot effectively detect the above targets in static stages through only the adjacent frame detection mode. Therefore, as shown in Figures 9a, 10a, 11a, 12a and 13a the dynamic target correct detection rates in each scene are not high, and the detection effects in scene 2 and scene 5 are the worst, mainly because the above scenes contain a large number of special dynamic targets. The detection effect in scene 4 is slightly lower. Due to the rainy weather, there is an additional systematic error in the LiDAR point cloud measurements, and the inter-frame distribution of the error is irregular. Therefore, it directly affects the calculation accuracy of the velocity and position variation between the paired point cloud clusters and indirectly affects the detection effect. In addition, because comparison method 1 requires setting the heuristic threshold as the benchmark to judge the target motion state, the default threshold has limited applicability to different environments and scenes, resulting in a low detection robustness. Therefore, the trend of the detection results, as shown by the red line in all subgraphs over time, generally fluctuates greatly, and the lines corresponding to different scenes greatly differ.

Comparison method 2 trains the dynamic target detection network in an unsupervised mode. Although this approach can successfully detect some dynamic targets with the above special motion states, it is limited by the network structure and cannot effectively judge the motion state of the occluded target, and the detection effect decreases as the laser ranging length increases. Therefore, as shown in Figures 9a, 10a, 11a, 12a and 13a, the dynamic target correct detection rates in each scene are also not high, but the detection results are generally better than those of comparison method 1, because there is no need to set the heuristic threshold. Among them, due to the serious occlusion problem between the dynamic targets in scene 2, the corresponding detection effect of this scene is the worst. For scene 4, since the detection network trained by this method directly detects the dynamic target in the single frame point cloud, the above absolute detection mode is sensitive to the quality of point cloud data. Therefore, when environmental factors lead to a decrease in point cloud quality, there are more error detection phenomena. This method has the best detection effect in scene 5. Since there are fewer people in this scene, the occlusion between the targets is less, and the LiDAR measurement performance is not affected by the illumination conditions, the trained detection network can effectively identify most stationary cars and judge them as dynamic targets. As shown in Figures 9b, 10b, 11b, 12b and 13b, the static target detection effect of comparison method 2 is relatively stable, but except for scene 5, many point clouds are generated by dynamic target scanning in the retained static point cloud in other scenes, which does not effectively improve the quality of the LiDAR point cloud data.

As shown in Figures 9a, 10a, 11a, 12a and 13a, for five different indoor and outdoor scenes, the dynamic target detection accuracy of our method is greater than 85%, and the trend of the line is relatively stable, indicating that our method has a high detection accuracy and robustness for dynamic targets. Among them, the missed detection frequency of dynamic targets is higher in scene 2 and scene 5 than in the other scenes, primarily because our method cannot effectively detect dynamic targets that are always in static states during the data acquisition phase. Compared with the reference benchmark of the above scenes, static vehicles or temporary stalls existing in the data acquisition process are regarded as dynamic targets, but these dynamic targets are not detected by our method. Although the above situation also occurs in the other scenes, the number of these targets is limited in those scenes, so the detection accuracy is not significantly impacted. In addition, because our method comprehensively judges the motion state of point cloud clusters by calculating the multi-dimensional position and geometric structure differences of adjacent frame paired point cloud clusters, the above relative detection mode is less affected by environmental factors and the multi-dimensional features ensure the robustness of the detection process. Therefore, in scene 4, our method still obtains effective detection results. As shown in Figures 9b, 10b, 11b, 12b and 13b, since our method comprehensively considers the multi-dimensional information to detect the target motion state and performs a secondary correction, it can be effectively applied to a situation where dynamic targets block each other and dynamic targets block static targets in the environment. Therefore, the static target error detection rate of our method in each scene is relatively stable and generally below 2%. As shown in Figure 13b, the static target error detection rate of our method is generally below 1.1% in scene 5, and the overall trend is stable, indicating that our method can still maintain a robust detection ability under weak illumination conditions. Because the static target base in the actual environment is large, when the static target error detection rate is low, its impact on the subsequent corresponding point cloud processing work can be ignored.

The reasons for the ups and downs of the change trend of each broken line in Figures 9–13 are analyzed as follows. Due to the large flow of people or vehicles in scene 1, scene 2, and scene 4, other targets will appear near the experimental platform at any time. The above targets will block part of the laser beam, resulting in the loss of more occluded other targets in the point clouds at this stage. The corresponding numbers of dynamic targets and static targets are greatly different from those at other times, which leads to the fact that the dynamic target correct detection rate and the static target error detection rate in Figures 9, 10 and 12 generally fluctuate greatly with time. Compared with the above three scenes, due to the small flow of people and vehicles in scene 5, the corresponding detection rate in Figure 13 is relatively stable with time, and the detection rate will change abruptly only in individual positions due to the change in the scanning field of view. As shown in Figure 6c, the environment in the first half of the data acquisition in scene 3 is relatively empty and secluded. When the experimental platform moves to the leisure area and the stadium in the second half, the corresponding detection rate in the second half of Figure 11 begins to fluctuate greatly with time due to the occlusion of other targets. In addition, due to the need to recover data acquisition equipment, the environment at the end of each data acquisition stage is relatively quiet and the dynamic target detection accuracy of the corresponding stage in each scene is generally relatively high.

After the above analysis of the limitations of the three methods and their detection performances in five experimental scenes, the detection effects of the three methods are quantitatively compared in Table 3. Compared with the two comparison methods, the average dynamic target correct detection rate of our method is 17.83% and 5.95% greater, the average static target error detection rate is 61.46% and 42.80% lower, the average detection efficiency is 27.60% and 10.75% higher, and the average real-time measurement index is 19.41% and 5.37% higher. The dynamic target correct detection rate is 92.41%, the static target error detection rate is 1.43%, and the detection efficiency is 0.0299 s. In addition, compared with the comparison methods, the average target motion state detection Accuracy, Precision, Recall, and F1-Measure of our method are also significantly improved. The Accuracy reaches 95.55%, indicating that the target motion state detection accuracy of our method is high. Precision and Recall reach 98.30% and 92.41%, respectively, indicating that our method has a strong dynamic target detection ability and also can effectively identify static targets. F1-Measure reaches 95.26%, indicating that the comprehensive detection ability of our method for dynamic targets and static targets is relatively balanced and has a strong robustness. In summary, our method has universal applicability to different experimental scenes. It can efficiently and accurately detect LiDAR dynamic targets and effectively improve the quality of LiDAR point clouds by screening dynamic point clouds.

In addition, considering that experimental scene 2 has a large range and contains rich targets, the point cloud collected in this scene is used as the test dataset to deeply test the influences of the volume factor, rate factor, and distribution position factor of the targets on the detection effect of our method. Based on the cloud cluster feature quantification methods in Section 2.3.1, the volume of the point cloud clusters, the distance between the center point of the point cloud clusters and the origin points, and the motion rate of the center point between the adjacent frame paired point cloud clusters are further calculated. Based on the above indicators, each point cloud cluster in the test dataset is divided into nine categories according to the rules shown in Table 4. The dynamic target in each category is regarded as a positive sample and the static target is regarded as negative sample. The target motion state detection Accuracy, Precision, Recall, and F1-Measure of different categories are calculated, as shown in Table 5.

Table 4. Point cloud clusters classification rules.

Classification Indications	Category 1	Category 2	Category 3
Volume factor (m^3)	[0, 2]	(2, 15)	[15, +∞]
Rate factor (m/s)	[0, 5]	(5, 10)	[10, +∞]
Distribution position factor (m)	[0, 15]	(15, 30)	[30, +∞]

Table 5. The detection effect of different categories in scene 2.

Classification Indications		Accuracy	Precision	Recall	F1-Measure
Volume factor	Category 1	94.46%	97.10%	89.04%	92.90%
	Category 2	94.96%	97.33%	90.06%	93.55%
	Category 3	95.09%	97.01%	90.73%	93.76%
Rate factor	Category 1	93.90%	96.81%	87.88%	92.13%
	Category 2	95.25%	97.05%	91.09%	93.97%
	Category 3	95.21%	97.11%	90.93%	93.92%
Distribution position factor	Category 1	95.34%	97.04%	91.32%	94.09%
	Category 2	94.80%	96.85%	90.14%	93.37%
	Category 3	92.82%	94.87%	87.03%	90.78%

As shown in the quantitative results of the detection effect in Table 5, the difference between the detection effects of the above nine categories is generally small and the detection effects of the three categories corresponding to different volume factors are similar. In the categories of rate factor, only the detection effect of the first category is slightly worse, and in the categories of distribution position factor, only the detection effect of the third category is worse. The reasons for the analysis are as follows. Our method extracts twelve features to construct the point cloud cluster classification feature system, which covers one-dimensional to three-dimensional, length to volume, and other features. When judging the motion state of the point cloud clusters, the above features play comprehensive roles. Therefore, even if there are abnormalities in the quantitative results of individual features, our method can still obtain robust detection results, so the corresponding detection effects of each category are generally less different. Except for the three-axis displacement feature, the remaining features in the feature system of our method are quantified by calculating the relative change degree. Therefore, the factors related to the geometric structure of the point cloud cluster will not affect the detection effect of our method. As a result, the detection effect between the categories corresponding to different volume factors is similar. Since our method cannot effectively detect the dynamic targets that always remain stationary during whole data acquisition, and these targets belong to the first category of the rate factor, the corresponding detection effect is slightly worse. The point cloud clusters contained in the third category of the distribution location factors are distributed in a range greater than 30 m from the origin. This category of target is usually generated by the residual laser beam after occlusion by many close-range targets. Because the distance between the laser scanning beam continues to expand with the scanning distance, the probability of an abnormal occurrence of the corresponding twelve features of these targets after quantification is generally large, resulting in poor detection results. However, due to the small number of these targets and their minimal role in point cloud inter-frame registration, point cloud model construction, and other related work, not too much energy needs to be spent to improve the detection effect of such targets. In summary, the detection effect of our method for LiDAR target motion state is less affected by factors such as volume and rate, and has a strong robustness.

4. Conclusions and Discussion

4.1. Conclusions

Aiming at the problem of current LiDAR dynamic target detection methods requiring heuristic thresholding, indirect computational assistance, supplementary sensor data, or postdetection, this paper proposes a LiDAR dynamic target detection method based on multidimensional features to detect the motion states of LiDAR point clouds with high efficiency and high precision. The method is analyzed and summarized using relevant experimental results.

- In this paper, 12 kinds of point cloud cluster features are extracted and quantified from the perspective of differences between the multidimensional positions and geometric structures of adjacent frame paired point cloud clusters. These features include uniaxial displacement, uniaxial span change degree, biaxial projection area change degree, point cloud number change degree, and other information, and are selected to ensure that the constructed point cloud cluster classification feature system is comprehensive. By constructing an a priori point cloud map and using the descriptor comparison method based on the voxel-based method, the motion state classification label is automatically assigned to the point cloud cluster in batches, avoiding human error. Based on the above point cloud cluster classification feature system and classification label quantization method, an experimental platform is built to collect data from real indoor and outdoor scenes to construct the model training dataset. To ensure the universal applicability of the LiDAR dynamic target detection model, the training set acquisition scene in this paper contains five representative indoor and outdoor environments.

- The training dataset of the proposed model contains many samples and the point cloud cluster classification feature system contains many classification features. Therefore, to account for the accuracy and efficiency of the LiDAR dynamic target detection model, the SCC−BSDR strategy is applied to the XGBoost training process. The test dataset is obtained from the UCI machine learning database and an ablation experiment is designed to verify the effectiveness of the above strategy. The results show that our algorithm is superior to the comparison algorithms when considering both classification accuracy and training efficiency. SCC−BSDR ensures that our method has a wider applicability and improves the average model training efficiency by 50.38%.
- Through model training, the optimal classification feature subset and corresponding XGBoost detection model are obtained. Considering the mechanical nature of machine learning and the influence of special dynamic targets on detection, a DBMV−SW strategy is proposed to correct the preliminary detection results of XGBoost twice. An ablation experiment is also designed to verify the effectiveness of this strategy. The results show that, after DBMV−SW correction, the number of dynamic target missed detections is reduced by 40.29%, the number of static target false detections is reduced by 14.34%, and the detection efficiency is reduced by 22.48%. The ratio of the accuracy improvement rate to the efficiency reduction rate is 1.59, indicating that this strategy sacrifices a small part of the operation efficiency but obtains a large improvement in accuracy.
- To comprehensively evaluate the detection effect of our method on LiDAR dynamic targets, two open-source LiDAR dynamic target detection methods based on the segmentation-based method are selected for comparison. The results show that the dynamic target correct detection rate of our method is 92.41%, the static target error detection rate is 1.43%, and the detection efficiency is 0.0299 s. Compared with those of the other two methods, the dynamic target correct detection rate of our method is 17.83% and 5.95% higher, the static target error detection rate is 61.46% and 42.80% lower, the detection efficiency is 27.60% and 10.75% higher, the real-time measurement index is 19.41% and 5.37% higher, and other comprehensive evaluation indexes are also significantly improved. Since our method comprehensively considers the multidimensional features between paired point cloud clusters to detect the motion states of point cloud clusters and constructs an optimal point cloud cluster classification feature subset and detection model, the detection accuracy and efficiency have significant advantages, and it has universal applicability for different scenes or conditions.
- Usually, LiDAR can meet most of the work requirements by collecting point clouds at the frequency of 10 Hz, and the corresponding point cloud inter-frame time interval is 0.1 s. At present, the average detection efficiency of our method for a frame of point cloud is 0.0302 s. Therefore, our method can be used as a point cloud preprocessing module for various practical works based on point cloud data. It is used to filter out dynamic point clouds to improve data quality, so as to ensure that the subsequent corresponding work results are more reliable.

- The point cloud double registration method proposed in this paper can register the center point sets of the adjacent frame point cloud clusters and obtain an effective pose relationship. It can be used alone to provide the initial value of the pose relationship for the SLAM registration algorithm based on the whole point cloud, thereby accelerating the convergence speed and improving accuracy. In addition, the SCC−BSDR proposed in this paper can also be transplanted to other multi-dimensional feature-based machine learning methods to screen optimal feature subsets, thereby improving data processing efficiency and ensuring excellent results.

4.2. Discussion

In this section, according to the experimental results and the corresponding conclusions, the advantages, shortcomings, and factors affecting the detection effect of our method and future research directions are discussed.

- Compared with other existing LiDAR dynamic target detection methods, our method does not require setting heuristic thresholds or using auxiliary processes such as plane projection and grid division. Furthermore, our method can directly detect the motion state of each point cloud cluster in the LiDAR point cloud. Since our method comprehensively detects the motion state of the target based on the multi-dimensional position and geometric structure difference between the adjacent frame paired point cloud clusters, and the preliminary detection results are corrected for the second time, the detection effect of our method is less affected by factors such as the volume, rate, and distribution position of the targets, and it has a universal applicability to different environments. Even under non-ideal conditions such as severe target occlusion, a complex scene structure, or rainy weather, our method can still maintain a strong robustness.
- In addition to the composition of the classification feature system, the secondary correction effect of the DBMV−SW strategy is a key factor that determines the final detection accuracy and efficiency. By analyzing the factors that can affect the above correction effect, three main factors are identified in this paper: the number of special dynamic targets, the frequency of motion state changes, the occlusion between targets, and the fitting function error. Specifically, the greater the number of dynamic targets with an intermittent static state or a first static and then moving state, or the greater the frequency of motion state changes, the more frequently the corresponding fitting labels and slope labels of these kinds of point cloud clusters will change. This will result in a greater probability of errors in the voting link in the secondary correction; this probability can be weakened by optimizing the voting strategy. The more serious the occlusion between the targets or the more common the occlusion phenomenon is, the more point cloud clusters that must be corrected or judged in depth after the voting process, increasing calculation costs. The error of the fitting function has little effect and can be weakened by optimizing the form of the fitting function; however, the first-order linear function can meet the current accuracy requirements and ensure a high data processing efficiency.
- Although our method achieves a high detection accuracy and efficiency, it is unable to effectively detect dynamic targets that remain stationary during whole data acquisition. To address the above problem, in the future research work, semantic information can be added to the point cloud cluster during the training process and dynamic targets can be eliminated by semantic label assistance, but the data processing workload of the scheme is significantly increased. In addition, a suitable scheme can be designed to correct the detection results of our method afterwards, but it can only meet post-application requirements. How to ensure its real-time performance needs to be further studied.
- With an increase in point cloud acquisition frequency, the detection efficiency of our method limits its application ability. When the point cloud is collected at the frequency of 20 Hz, the corresponding point cloud inter-frame time interval is 0.05 s. Although

it is greater than the time required for the detection of our method, if it is used as a preprocessing module for other work, the overall work may find it difficult to meet the real-time performance. To address the above problem, more efficient machine learning methods or an improved fitting function form and voting process in DBMV−SW are considered in future research work, so as to improve the computational efficiency of our method.

- At present, our method can be applied to non-ideal conditions such as weak illumination or rainy weather. However, for extreme weather conditions such as haze days, heavy rain days, and heavy snow days, due to the high density of water or impurities in the atmosphere, the measurement performance of LiDAR is greatly affected, which, in turn, greatly affects the quality of the original point cloud, which eventually leads to a decrease in the detection accuracy of our method or even making it unusable. In future research work, for the case of less influence, it is considered to add an atmospheric correction module to compensate for the measurement error, thereby improving the quality of the point cloud. For the case of large influence, we consider using Radio Detection and Ranging (Radar) with a stronger penetration to collect data, and use our method to fuse and solve, so as to obtain effective dynamic target detection results.

Author Contributions: Conceptualization, J.G. and A.X.; data curation, J.G., C.W. and Z.S.; formal analysis, J.G., X.S. and C.W.; funding acquisition, A.X.; investigation, J.G.; methodology, J.G. and X.S.; software, J.G., C.W. and Z.S.; validation, J.G. and Z.S.; writing—original draft preparation, J.G.; writing—review and editing, J.G., A.X., X.S., C.W. and Z.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded in part by the National Natural Science Foundation of China (No. 42074012), and in part by the Liaoning Revitalization Talents Program (Nos. XLYC2002101 and XLYC2008034).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Durrant, W.H.; Bailey, T. Simultaneous localization and mapping: Part I. *IEEE Robot. Autom. Mag.* **2006**, *13*, 99–110. [CrossRef]
2. Xu, X.; Zhang, L.; Yang, J.; Cao, C.; Wang, W.; Ran, Y.; Tan, Z.; Luo, M. A review of multi-sensor fusion slam systems based on 3D LIDAR. *Remote Sens.* **2022**, *14*, 2835. [CrossRef]
3. Zhang, J.; Zhang, X.; Shen, X.; Wu, J.; Li, Y. A Lidar SLAM based on Improved Particle Filter and Scan Matching for Unmanned Delivery Robot. *J. Phys. Conf. Series. IOP Publ.* **2023**, *2506*, 12009–12018. [CrossRef]
4. Inostroza, F.; Parra-Tsunekawa, I.; Ruiz-del-Solar, J. Robust Localization for Underground Mining Vehicles: An Application in a Room and Pillar Mine. *Sensors* **2023**, *23*, 8059. [CrossRef]
5. Ricciardelli, E.; Di Paola, F.; Cimini, D.; Larosa, S.; Mastro, P.; Masiello, G.; Serio, C.; Hultberg, T.; August, T.; Romano, F. A Feedforward Neural Network approach for the detection of optically thin cirrus from IASI−NG. *IEEE Trans. Geosci. Remote Sens.* **2023**, *61*, 4104217. [CrossRef]
6. Wan, E.; Zhang, Q.; Li, L.; Li, X.; Liu, Y. The online in situ detection of indoor air pollution via laser induced breakdown spectroscopy and single particle aerosol mass spectrometer technology. *Opt. Lasers Eng.* **2024**, *174*, 107974. [CrossRef]
7. Poux, F.; Mattes, C.; Selman, Z.; Kobbelt, L. Automatic region-growing system for the segmentation of large point clouds. *Autom. Constr.* **2022**, *138*, 104250. [CrossRef]
8. Wu, L.; Wang, G.; Hu, Y. Iterative closest point registration for fast point feature histogram features of a volume density optimization algorithm. *Meas. Control* **2020**, *53*, 29–39. [CrossRef]
9. Guo, F.; Zheng, W.; Lian, G.; Zhang, X.; Luo, L.; Wu, Y.; Guo, P. A point cloud registration method based on multiple-local-feature matching. *Optik* **2023**, *295*, 171511. [CrossRef]
10. Karam, S.; Lehtola, V.; Vosselman, G. Strategies to integrate IMU and LiDAR SLAM for indoor mapping. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2020**, *1*, 223–230. [CrossRef]

11. Giveki, D. Robust moving object detection based on fusing Atanassov's Intuitionistic 3D Fuzzy Histon Roughness Index and texture features. *Int. J. Approx. Reason.* **2021**, *135*, 1–20. [CrossRef]
12. Yao, W.; Hinz, S.; Stilla, U. Extraction and motion estimation of vehicles in single-pass airborne LiDAR data towards urban traffic analysis. *ISPRS J. Photogramm. Remote Sens.* **2011**, *66*, 260–271. [CrossRef]
13. Shi, H.; Lin, G.; Wang, H.; Hung, T.Y.; Wang, Z. Spsequencenet: Semantic segmentation network on 4d point clouds. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020. [CrossRef]
14. Pfreundschuh, P.; Hendrikx, H.F.C.; Reijgwart, V.; Dube, R.; Siegwart, R.; Cramariuc, A. Dynamic object aware lidar slam based on automatic generation of training data. In Proceedings of the 2010 IEEE International Conference on Robotics and Automation, Xi'an, China, 30 May–5 June 2021. [CrossRef]
15. Li, S.; Chen, X.; Liu, Y.; Dai, D.; Stachniss, C.; Gall, J. Multi-scale interaction for real-time lidar data segmentation on an embedded platform. *IEEE Robot. Autom. Lett.* **2021**, *7*, 738–745. [CrossRef]
16. Chen, G.; Wang, B.; Wang, X.; Deng, H.; Wang, B.; Zhang, S. PSF−LO: Parameterized semantic features based LiDAR odometry. In Proceedings of the 2010 IEEE International Conference on Robotics and Automation, Xi'an, China, 30 May–5 June 2021. [CrossRef]
17. Chen, X.; Milioto, A.; Palazzolo, E.; Giguere, P.; Behley, J.; Stachniss, C. Suma++: Efficient lidar-based semantic slam. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019. [CrossRef]
18. Qian, C.; Xiang, Z.; Wu, Z.; Sun, H. RF−LIO: Removal-first tightly-coupled lidar inertial odometry in high dynamic environments. *arXiv* **2022**, *6*, 9463–9471. [CrossRef]
19. Kim, G.; Kim, A. Remove, then revert: Static point cloud map construction using multiresolution range images. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24–30 October 2020. [CrossRef]
20. Chen, X.; Li, S.; Mersch, B.; Wiesmann, L.; Gall, J.; Behley, J.; Stachniss, C. Moving object segmentation in 3D LiDAR data: A learning-based approach exploiting sequential data. *IEEE Robot. Autom. Lett.* **2021**, *6*, 6529–6536. [CrossRef]
21. Schauer, J.; Nüchter, A. The peopleremover—Removing dynamic objects from 3-d point cloud data by traversing a voxel occupancy grid. *IEEE Robot. Autom. Lett.* **2018**, *3*, 1679–1686. [CrossRef]
22. Lim, H.; Hwang, S.; Myung, H. ERASOR: Egocentric ratio of pseudo occupancy-based dynamic object removal for static 3D point cloud map building. *IEEE Robot. Autom. Lett.* **2021**, *6*, 2272–2279. [CrossRef]
23. Zhou, Z.; Feng, X.; Di, S.; Zhou, X. A LiDAR Mapping System for Robot Navigation in Dynamic Environments. In Proceedings of the 2023 IEEE Intelligent Vehicles Symposium, Anchorage, AK, USA, 4–7 June 2023. [CrossRef]
24. Chen, T.; Guestrin, C. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794. [CrossRef]
25. Nguyen, V.Q.; Tran, V.L.; Nguyen, D.D.; Sadiq, S.; Park, D. Novel hybrid MFO−XGBoost model for predicting the racking ratio of the rectangular tunnels subjected to seismic loading. *Transp. Geotech.* **2022**, *37*, 100878–100891. [CrossRef]
26. Srikanth, B.; Papineni, V.L.S.; Sridevi, G.; Indira, D.; Radhika, K.S.R.; Syed, K. Adaptive XGBOOST hyper tuned meta classifier for prediction of churn customers. *Intell. Autom. Soft Comput.* **2022**, *33*, 21–34. [CrossRef]
27. Poornima, R.; Elangovan, M.; Nagarajan, G. Network attack classification using LSTM with XGBoost feature selection. *J. Intell. Fuzzy Syst.* **2022**, *43*, 971–984. [CrossRef]
28. Wang, F.; Yu, J.; Liu, Z.; Kong, M.; Wu, Y. Study on offshore seabed sediment classification based on particle size parameters using XGBoost algorithm. *Comput. Geosci.* **2021**, *149*, 104713. [CrossRef]
29. Lin, N.; Zhang, D.; Feng, S.; Ding, K.; Tan, L.; Wang, B.; Chen, T.; Li, W.; Dai, X.; Pan, J.; et al. Rapid Landslide Extraction from High-Resolution Remote Sensing Images Using SHAP−OPT−XGBoost. *Remote Sens.* **2023**, *15*, 3901. [CrossRef]
30. Zhang, L.; Guo, Z.; Tao, Q.; Ye, J. XGBoost-based short-term prediction method for power system inertia and its interpretability. *Energy Rep.* **2023**, *9*, 1458–1469. [CrossRef]
31. Du, X.; Chen, Y.; Zhang, X.; Guo, Y. Study on Feature Engineering and Ensemble Learning for Student Academic Performance Prediction. *Int. J. Adv. Comput. Sci. Appl.* **2022**, *13*, 495–502. [CrossRef]
32. Levinson, J.; Thrun, S. Unsupervised calibration for multi-beam lasers. In Proceedings of the Experimental Robotics: The 12th International Symposium on Experimental Robotics, Berlin, Germany, 12–17 March 2014; Available online: https://link.springer.com/chapter/10.1007/978-3-642-28572-1_13 (accessed on 12 August 2023).
33. Bezet, O.; Cherfaoui, V. Time error correction for laser range scanner data. In Proceedings of the 2006 9th International Conference on Information Fusion, Las Vegas, NV, USA, 10–13 July 2006. [CrossRef]
34. Lee, Y.W. Statistical filtering and prediction. In *Il Nuovo Cimento Series 10*; SOCIETÀ ITALIANA DI FISICA; Italian Physical Society: Varenna, Italy, 1959; Volume 13, pp. 430–454. [CrossRef]
35. Hong, S.; Ko, H.; Kim, J. VICP: Velocity updating iterative closest point algorithm. In Proceedings of the 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 3–7 May 2010. [CrossRef]
36. Lin, C.; Liu, H.; Wu, D.; Gong, B. Background point filtering of low-channel infrastructure-based LiDAR data using a slice-based projection filtering algorithm. *Sensors* **2020**, *20*, 3054. [CrossRef] [PubMed]

37. Li, Y.; Zhang, R.; Shao, X.; Xu, Y. Improved Filtering and Hole Filling Algorithm for the Point Cloud of Rotor Surface Based on PCL. In Proceedings of the 2021 IEEE International Conference on Power Electronics, Computer Applications (ICPECA), Shenyang, China, 22–24 January 2021. [CrossRef]
38. Patiño, H.; Zapico, P.; Rico, J.C.; Valino, G.; Fernandez, P. A Gaussian filtering method to reduce directionality on high-density point clouds digitized by a conoscopic holography sensor. *Precis. Eng.* **2018**, *54*, 91–98. [CrossRef]
39. El, H.S.; Merras, M. Improvement of 3D reconstruction based on a new 3D point cloud filtering algorithm. *Signal Image Video Process* **2023**, *17*, 2573–2582. [CrossRef]
40. Zhao, Q.; Gao, X.; Li, J.; Luo, L. Optimization algorithm for point cloud quality enhancement based on statistical filtering. *J. Sens.* **2021**, *2021*, 7325600. [CrossRef]
41. Hu, X.; Yan, L.; Xie, H.; Dai, J.; Zhao, Y.; Su, S. A novel lidar inertial odometry with moving object detection for dynamic scenes. In Proceedings of the 2022 IEEE International Conference on Unmanned Systems (ICUS), Guangzhou, China, 28–30 October 2022. [CrossRef]
42. Besl, P.J.; McKay, N.D. Method for registration of 3-D shapes. In *Sensor Fusion IV: Control Paradigms and Data Structures. Spie*; Robotics '91: Boston, MA, USA, 1991; Volume 16, pp. 586–606. [CrossRef]
43. Kim, J.; Yoon, S.; Choi, T.; Sull, S. Unsupervised Video Anomaly Detection Based on Similarity with Predefined Text Descriptions. *Sensors* **2023**, *23*, 6256. [CrossRef]
44. Chen, Y.; Lu, Y. Performing 3D similarity transformation by robust total least squares. *Acta Geod. Et Cartogr. Sin.* **2012**, *41*, 715–722.
45. Martínez-Otzeta, J.M.; Rodríguez-Moreno, I.; Mendialdua, I.; Sierra, B. Ransac for robotic applications: A survey. *Sensors* **2022**, *23*, 327. [CrossRef]
46. Zhang, J.; Singh, S. Low-drift and real-time lidar odometry and mapping. *Auton. Robot.* **2017**, *41*, 401–416. [CrossRef]
47. Liu, Y.; Zhang, W.; Li, F.; Zuo, Z.; Huang, Q. Real-time lidar odometry and mapping with loop closure. *Sensors* **2022**, *22*, 4373. [CrossRef] [PubMed]
48. Zhao, D.; Ji, L.; Yang, F. Land Cover Classification Based on Airborne Lidar Point Cloud with Possibility Method and Multi-Classifier. *Sensors* **2023**, *23*, 8841. [CrossRef] [PubMed]
49. Li, Y.; Fan, W.; Song, L.; Liu, S. Combining emerging hotspots analysis with XGBoost for modeling pedestrian injuries in pedestrian-vehicle crashes: A case study of North Carolina. *J. Transp. Saf. Secur.* **2023**, *15*, 1203–1225. [CrossRef]
50. Sedgwick, P. Spearman's rank correlation coefficient. *BMJ* **2014**, *349*, 7327–7330. [CrossRef] [PubMed]
51. Binsaeed, K.A.; Hafez, A.M. Enhancing Intrusion Detection Systems with XGBoost Feature Selection and Deep Learning Approaches. *Int. J. Adv. Comput. Sci. Appl.* **2023**, *14*, 1084–1098. [CrossRef]
52. Ma, J.; Yuan, G.; Guo, C.; Gang, X.; Zheng, M. SW-UNet: A U-Net fusing sliding window transformer block with CNN for segmentation of lung nodules. *Front. Med.* **2023**, *10*, 1273441–1273457. [CrossRef]
53. Amarnath, B.; Balamurugan, S.; Alias, A. Review on feature selection techniques and its impact for effective data classification using UCI machine learning repository dataset. *J. Eng. Sci. Technol.* **2016**, *11*, 1639–1646.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

sensors

MDPI

Article

SAE3D: Set Abstraction Enhancement Network for 3D Object Detection Based Distance Features

Zheng Zhang [1], Zhiping Bao [1], Qing Tian [1,*] and Zhuoyang Lyu [2]

[1] School of Information, North China University of Technology, Beijing 100144, China; zhangzheng@ncut.edu.cn (Z.Z.); wiskey@mail.ncut.edu.cn (Z.B.)

[2] School of Information, Brown University Computer Science and Applied Math, Providence, RI 02912, USA; zhuoyang_lyu@brown.edu

* Correspondence: tianqing@ncut.edu.cn

Abstract: With the increasing demand from unmanned driving and robotics, more attention has been paid to point-cloud-based 3D object accurate detection technology. However, due to the sparseness and irregularity of the point cloud, the most critical problem is how to utilize the relevant features more efficiently. In this paper, we proposed a point-based object detection enhancement network to improve the detection accuracy in the 3D scenes understanding based on the distance features. Firstly, the distance features are extracted from the raw point sets and fused with the raw features regarding reflectivity of the point cloud to maximize the use of information in the point cloud. Secondly, we enhanced the distance features and raw features, which we collectively refer to as self-features of the key points, in set abstraction (SA) layers with the self-attention mechanism, so that the foreground points can be better distinguished from the background points. Finally, we revised the group aggregation module in SA layers to enhance the feature aggregation effect of key points. We conducted experiments on the KITTI dataset and nuScenes dataset and the results show that the enhancement method proposed in this paper has excellent performance.

Keywords: 3D object detection; distance features; SA layer enhancement

1. Introduction

With the development of unmanned driving and other technologies, understanding 3D scenes based on the point cloud has become a popular research topic. Compared to traditional images, point cloud data have unique advantages. The strong penetration of LiDAR makes the point cloud less susceptible to external factors such as weather and light. However, point clouds are also characterized by sparseness and disorder, and the reflectivity of LiDAR decreases as the measurement distance increases. This leads to poor characterization of objects at a distance, causing a drop in detection accuracy. How to deal with these characteristics of point clouds has become the key to improving accuracy in 3D detection tasks.

In recent years, to efficiently utilize the information provided by the point cloud, researchers have proposed a number of schemes, as shown in Figure 1. These are mainly divided into two types based on different processing methods:

(a) Grid-based methods, which partition the sparse points into regular voxel or pillar grids, and process them through 3D or 2D convolutional networks.

(b) Point-based methods, which directly perform feature learning on point sets with SA which are most often utilized to sample the key points and aggregate features.

Compared to the point-based methods, the grid-based methods increase the computational speed of network inference, but also cause information loss during the voxelization. Therefore, to ensure the full utilization of information in point sets, a point-based methods enhancement network is proposed in this paper.

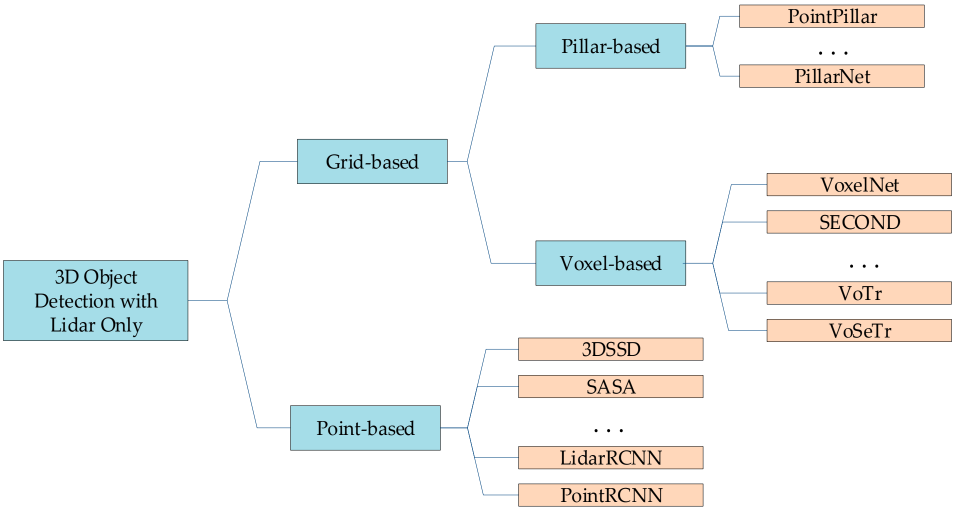


Figure 1. Overview of related work.

The core of the point-based 3D object detection methods is the SA layer, which was first proposed by Qi et al. [1]. In prior research, the SA layer has been revised using many methodologies, and how to fully utilize the information of each point and reduce inference time has become a priority in point-based methods. In 3DSSD [2], to speed up the inference, researchers first adopted a 3D single stage object detector and proposed a feature-based farthest point sample module (F-FPS). This module utilizes the feature information of the point sets to sample key points in order to maintain adequate interior points of different foreground instances. SASA [3] uses a semantic-segmentation-based farthest point sample module (S-FPS), which utilizes point cloud features distinguish the foreground points from the background points through a small semantic segmentation module to better access key points. However, the point cloud features used by these algorithms only utilize the raw features of the point cloud, i.e., the reflectivity and 3D coordinates that reveal spatial information of each point in the point cloud, and distance characteristics are not taken into consideration. In the actual measurement, due to the attenuation characteristics of LiDAR and the limitation of the observation angle, the reflectivity of the measured point decreases as the object moves further away, and the projection of the object in the point cloud also decreases.

Therefore, based on the distance characteristics related to the point cloud, we propose three feature enhancement modules to more efficiently utilize the semantic information contained within the point cloud. Firstly, we propose the initial feature fusion module, in which the distance feature is extracted from the point cloud and incorporated with the raw features of each point. Secondly, we introduce a key point feature enhancement module. During the group aggregation in SA, the self-characterization of the key points will be weakened, but it is crucial for distinguishing whether the key point is a foreground or background point. Therefore, after each sampling aggregation, the multi-attention mechanism is used to strengthen the features of key points and fuse them with the aggregated features. Finally, to enhance the effect of group aggregation in SA, we revised the original grouping module. In the original module, multiple points nearest to the key points are taken to participate in feature aggregation after sampling over the key points. However, only the spatial location is considered, which may result in features belonging to different categories being mixed together during the aggregation process. This leads to a decrease in the performance of the semantic segmentation module before S-FPS, which in turn degrades the sampling effect of S-FPS. Therefore, we optimize the grouping module by selecting the points with the closest features as the aggregation points from among multiple points closest to the key points.

In summary, the main contributions of this article are summarized as follows:

- We propose a key points self-features enhancement module to enhance the self-features of the key points. In this module, we introduce the multi-attention mechanisms to enhance the raw features and distance features to retain the semantic information of the key points as much as possible during each SA layer.
- We propose an initial feature fusion module to extract the distance features of the point cloud and fuse the distance features into the raw features of the point sets. This module makes the features of the distant points more significant and thus improves the detection accuracy of the distant instances.
- We revise the group aggregation module in the set abstraction. We make a second selection after the first selection of points within a fixed distance around the key point. In second selection, we take the features into account to enhance the sampling effect of S-FPS.

2. Related Work

Since the growing data on point clouds bring huge challenges to existing point cloud processing networks, it is important to compress the point cloud before processing it. Different compression algorithms used may affect the subsequent detection effect. For example, Sun X et al. [4] optimized the processing of large-scale point cloud data and their algorithm [5] further streamlines the network for point cloud processing. The algorithm [6] makes the spatial distribution of the compressed point cloud more similar to the original point cloud, which is very useful for subsequent point cloud processing.

The point cloud compression algorithms mentioned above play a significant role in the point cloud detection algorithms we will discuss next.

2.1. Grid-Based Methods

Grid-based methods are mainly divided into two categories: voxel-based methods and pillar-based methods. In voxel-based methods, an irregular point cloud is first con-verted into regular voxels, which are then fed into the network. Voxel-Net [7] is a pioneering network that converts point cloud into voxels, and then utilizes 3D convolutional networks to predict 3D bounding boxes. Yan et al. [8] proposed 3D sparse convolution, which reduces the computation of traditional 3D convolution and greatly improves the detection efficiency of voxel-based detection networks. Voxel-Transformer [9] and Voxel Set Transformer [10] introduce modules such as Transformer [11] and Set Transformer [12], respectively, on the basis of voxels to improve the detection accuracy. SFSS-Net [13] is a unique algorithm to filter background points before the voxelization to reduce computational complexity. Pillar-based methods such as Point Pillars [14] divide the space into regular pillars, which are compressed and then fed into a 2D convolutional network, greatly increasing the network inference speed. Pillar Net [15] uses a sparse convolutional-based encoder network for spatial feature learning, and the Neck module for high-level and low-level feature fusion to improve the accuracy of pillar-based detection methods. Pillar Next [16] first compares different local point aggregators (pillar, voxel and multi-view fusion) from the perspective of computational budget allocation. Research shows that pillars can achieve better performance compared to voxels. Grid-based methods lose more semantic information in the process of converting an irregular point cloud into regular voxels or pillars. This may lead to poor performance in the final detection accuracy.

2.2. Point-Based Methods

Point-based methods generally perform feature extraction directly on the point sets. This approach obtains key points and aggregates points around them by means of sampling and group aggregation for feature extraction. Point-based methods were first proposed by Qi et al. [17] and later improved and refined by Qi et al. [1]. Shi et al. [18] first proposed to extract the foreground points by segmentation and utilize the features of these points for the bounding box regression to improve the detection accuracy. Yang et al. [2] utilized one-stage detection to improve the inference speed and proposed the F-FPS, to make the sampled key points closer to the foreground instances. SASA [3] is used to predict scores of each point by a small semantic segmentation module to make abstracted point sets focus on object areas. Chen et al. [19] introduced density information from point clouds using the Multilayer Perceptron (MLP) and integrated it with features extracted by grouping operations in the point-based method.

Since point-based 3D object detection is directly processing the point sets, point-based methods result in high computational consumption and long network inference time. However, relative to the voxel-based methods, point-based methods can maximize the retention of the semantic information of the point cloud and achieve higher detection accuracy. Therefore, this paper adopts the point-based object detection network and aims to utilize the original information of the point cloud more efficiently.

3. Proposed Methods

In this section, we will introduce in detail the network architecture of the SAE3D proposed in this paper. This enhancement network consists of three main parts: an initial feature fusion module, a key points self-features enhancement (KSFE) module and a revised group aggregation (RGA) module.

As shown in Figure 2, the overall architecture is a one-stage point-based 3D object detection network. Firstly, we define the raw points fed into the network as P; the initial feature fusion module extracts the distance features and integrates initial features of each point in P. After integration, we feed P into the backbone, which contains three SA layers, and we refer to the input of each SA layer as P_1. In SA layers, we first sample the key points K from P_1, and then we feed K into the key points feature enhancement module to enhance the self-features of K. After enhancement, we obtain K_1. Finally, the revised group aggregation module is used to aggregate the points around K_1 to obtain the aggregated key points K_2. K_2 is the final output of each SA layer.

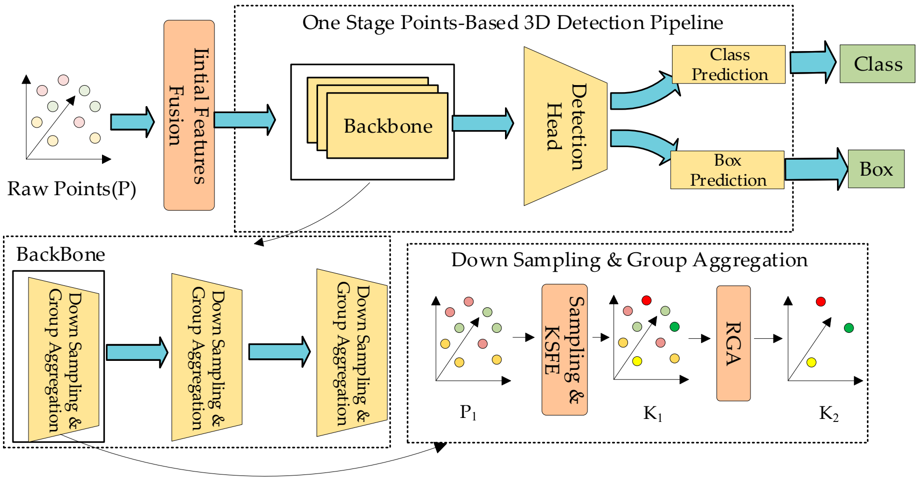

One Stage Points-Based 3D Detection Pipeline
Raw Points(P)
Iintial Features Fusion
Backbone
Detection Head
Class Prediction
Box Prediction
Class
Box
BackBone
Down Sampling & Group Aggregation
Down Sampling & Group Aggregation
Down Sampling & Group Aggregation
Down Sampling & Group Aggregation
P1
Sampling & KSFE
K1
RGA
K2

Figure 2. Overall flowchart. The raw point cloud P goes through the initial feature fusion module to get P_1, P_1 is input to the backbone, backbone consists of three SA (set abstraction) layers. P_1 is first put through the down sampling and then through the KSFE (key points self-feature enhancement module) to get K_1, and finally through the RGA (revised group aggregation module) to get K_2.

After the backbone is complete, to improve the prediction accuracy, this paper adopts the bounding box prediction mechanism in Vote-Net [20] to predict the bounding box similarly to SASA [3].

We will explain each module in detail below.

3.1. Initial Features Fusion Module

Before the SA layers, we utilize the initial features fusion module to extract the distance features and integrate these with features of the raw point sets. The relevant features of the raw point cloud are very sensitive to the measurement distance. In the actual measurement, as the distance increases, the reflectivity of LiDAR decreases, which leads to the problem that the features of the long-distance points are not obvious and thus reduce the detection accuracy. Therefore, we believe that distance features are very important for improving target detection accuracy.

3.1.1. Distance Features

Traditional algorithms often involve calculations such as squares and roots when calculating distance. This costs a lot of computational resources if we directly let distance represent the distance feature of each point in the point clouds. Therefore, our distance feature is defined as follows:

$$DF_p = \frac{|x_p| + |y_p| + |z_p|}{Scale} \qquad p \in P \tag{1}$$

where P is the raw point set, DF_p and (x_p, y_p, z_p) represent the distance feature and the coordinates of the p, respectively. *Scale* is the scaling factor. We utilize the sum of absolute values of the three-axis coordinates of p to represent the distance of a point. The *Scale* will be set in the experiment.

3.1.2. Feature Fusion

We process the initial feature fusion as shown in Figure 3. Since the reflectivity of each point decreases with the increase of the measurement distance, we adopt the approach of adding distance features with the initial features of the point cloud to strengthen the representation of long-distance points. Finally, we perform fusion operations on the coordinates and related features of the point cloud through the splicing operation. However, these features have not been processed enough, so we use the Multilayer Perceptron (MLP) to further extract the depth features.

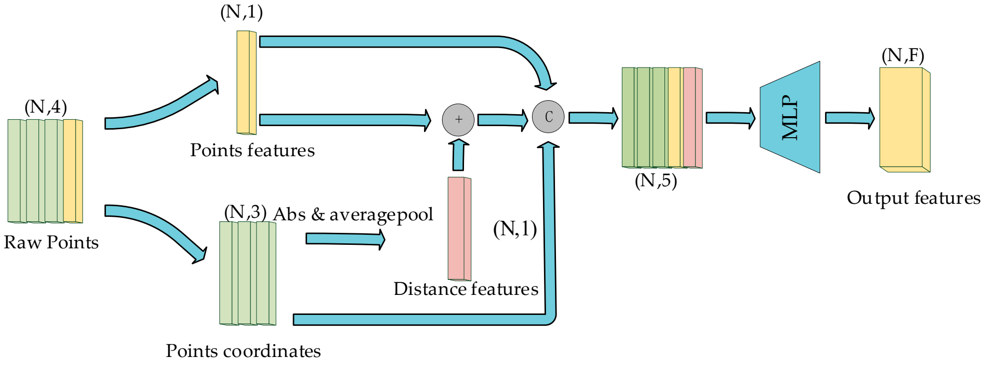


Figure 3. Initial features fusion module (where N stands for the number of input point clouds and F stands for the number of feature layers for each point in the output. "C" represents the stitching operation and "+" represents the numerical summing operation.).

3.2. Key Points Features Enhancement Module

In SA layers, the key points obtained from the sampling will undergo feature aggregation with the surrounding points, and the self-features of the key points will be diminished after aggregation with max or average pooling. However, each key point has its own unique features in the point cloud data, and these features contain important information included where the key point is located in the point cloud and what kind of object the key point represents. However, the feature aggregation will cause the loss of such information.

Therefore, we propose a key points self-feature enhancement module as shown in Figure 4, which enhances the distance features and raw features of the key points, integrating them into the aggregated features.

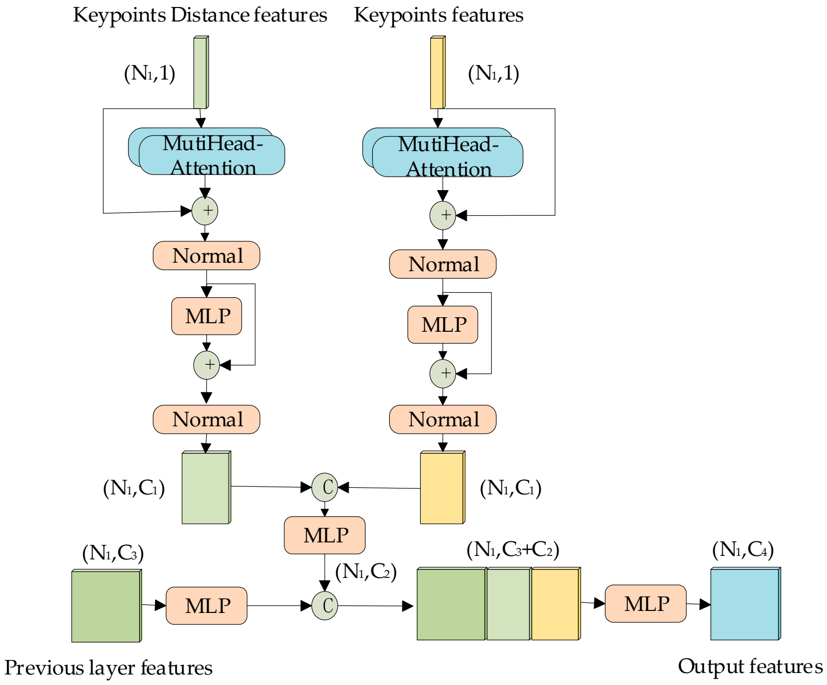


Figure 4. Key points self-features enhancement module (where N_1 is the number of key points after sampling, and C_i is the number of feature channels in each stage. "C" stands for the stitching operation and "+" stands for the numerical summing operation.).

Feature Enhancement Module

In order to make the self-features of the key points distinctive, we adopt the multi-attention mechanism to enhance the distance features and raw features of the key points. The features are strengthened through the multi-head self-attention mechanism; the self-attention algorithm essentially uses matrix multiplication to calculate the relationship between each patch and the other patches in the query. The specific formulas are as follows:

$$Attention(Q,K,V) = Softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2}$$

$$Q = F \times W_q \tag{3}$$

$$K = F \times W_k \tag{4}$$

$$V = F \times W_v \tag{5}$$

where F is the self-features of the key points, W_q, W_k and W_v are the learnable weight matrices. Equations (3)–(5) represent that F obtains Q, K, and V through three separate MLPs. After obtaining Q, K, and V, we use Equation (2) to finally obtain the attention features. After that we employ the splicing method to combine them together. Finally, we carry out the integration of the aggregated features of the key points with their self-features using the MLP to accomplish the enhancement of self-features of key points.

3.3. Revised Group Aggregation Module

In the process of sampling key points, we follow the S-FPS and D-FPS combined sampling strategy, which is similar to that of SASA [3]. A small semantic segmentation module is adopted in the network structure to compute the classification score for each point to distinguish between foreground and background points in the point cloud. The input features to the segmentation network are those obtained from grouping of the point sets. In the general grouping operation, the selection of points used for aggregation around the key points only considers the spatial location from the key points, not taking into account the feature distance from the key points. In this paper, it is argued that this aggregation operation diminishes the borderlines of the different instances and reduces the effectiveness of the segmentation module in predicting the classification scores of each point in the network, thus affecting the sampling performance of S-FPS. To avoid these problems, we perform a second selection after selecting points within a certain distance from each key point. In the second selection, we introduce the feature distance to ensure that the features of the selected points are similar to those of the key point. By doing so, we can enhance the performance of the segmentation in this network.

Group Aggregation Method

The particular operation is shown in Figure 5. First, we select N points as a point set P_N within the sphere with radius R around the key point, and calculate the feature distance D_f between the points and the key points, which we define as follows:

$$D_f = \left| f_{keypoints} - f_n \right| \qquad n \in N \tag{6}$$

where $f_{keypoints}$ and f_n separately represent the features of the key points and the features of the points around the key points. Before calculation, these features will go through a simple MLP to ensure that the features channel is one-dimensional. After obtaining D_f, we select the N_k points with the smallest D_{fk} (k = 1, 2, . . ., N_k) in P_N as a point set P_{Nk}. The P_{Nk} will be used for subsequent features aggregation. In this way, we further strengthen the semantic information of the key points. This can help S-FPS to better distinguish the foreground points from the background points before sampling.

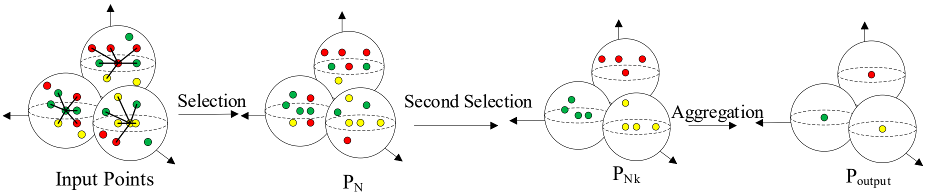


Figure 5. Revised group aggregation module (points with similar colors in the figure represent similar features, P_N is the point obtained by the first selection around the key point, P_{Nk} is the point obtained by the second selection, and P_{output} is the final point output after group aggregation).

3.4. Prediction Head

The overall architecture in this paper consists of three SA layers with a bounding box prediction network. Similarly, our bounding box prediction network adopts the bounding box prediction mechanism from Vote-Net [20]. The voting point indicating the center of mass of the corresponding object is first computed from the candidate point features, and then the points in the vicinity of each voting point are aggregated to estimate the bounding box of the detected target.

3.5. Loss

The loss function in SAE3D is inherited from SASA [3]. The overall loss function is expressed as follows:

$$L = L_v + L_c + L_r + L_{seg} \tag{7}$$

where L_c and L_r are the losses for the classification and regression, L_v is the loss generated when calculating the vote in the point voting head proposed in Vote-Net [20]. L_{seg} is the total segmentation loss proposed in SASA [3].

L_c and L_r are the traditional losses for object detection. They can help the network better predict the bounding box and classification of the object to be detected. L_v mainly serves to predict the center point of the object to improve the accuracy of bounding box prediction. L_{seg} mainly serves to perform semantic segmentation before the S-FPS to better differentiate between foreground and background points. This can improve the sampling capability of the S-FPS. Therefore, we adopt these loss functions to better train our model.

4. Experiment

4.1. Datasets

The network we proposed is validated on the KITTI dataset and nuScenes dataset.

4.1.1. KITTI Dataset [21]

The KITTI dataset is a widely used public dataset in the field of computer vision, which is primarily utilized to study and evaluate tasks such as autonomous driving, scene understanding, and target detection. The dataset is based on the streets of Karlsruhe, Germany, and comprises a wide range of urban driving scenarios. The KITTI dataset has become the mainstream standard for 3D object detection in traffic scenes due to its provision of data from real-world scenarios with a high level of realism and representative value.

In the original KITTI dataset, each sample comprises multiple consecutive frames of point cloud data. In our experiment, a total of 7481 point clouds are included along with 3D bounding boxes for training purposes, and 7581 samples are allocated for testing. We adopt a general setup where the training samples are further subdivided into 3712 training samples and 3769 testing samples. Our experimental network is trained on the training samples and validated on the testing samples.

4.1.2. NuScenes Dataset [22]

The nuScenes dataset is one of the more challenging datasets for autopilot research. It comprises 380,000 LiDAR scans from 1000 scenes and is labeled with up to 10 object categories, including 3D bounding boxes, object velocities, and attributes. The detection range is 360 degrees. The nuScenes dataset is evaluated using metrics such as the commonly used mean Average Precision (mAP) and the novel nuScenes Detection Score (NDS), which reflects the overall quality of measurements across multiple domains.

When transferring the nuScenes dataset, we combine LiDAR points from the current key frame and previous frames within 0.5 s, which involves up to 400 k LiDAR points in a single training sample. We then reduce the number of input LiDAR points. Specifically, we voxelize the point cloud from the key frame as well as the stacked previous frames with pixel sizes of (0.1 m, 0.1 m, 0.1 m), then randomly select 16,384 and 49,152 voxels from the key frame and the previous frames, respectively. For each selected voxel, we randomly select one internal LiDAR point. A total of 65,536 points were fed into the network with 3D coordinates, reflectivity, and timestamps.

4.1.3. Evaluation Indicators

In the experiment on the KITTI dataset, two precision metrics are used. One is the 11-point interpolated average precision (AP) proposed by Gerard et al. [23], and the other is the average precision $AP|_{R40}$ for 40 recalled positions proposed by Simonelli et al. [24]. The

Intersection over Union (IoU) threshold for all precision calculations is 0.70. The specific formulas of $AP|_R$ are as follows:

$$AP\bigg|R = \frac{1}{|R|}\sum_{r\in R}\rho interp(r) \tag{8}$$

$$\rho interp(r) = \max_{r':r'\geq r} p(r') \tag{9}$$

where $p(r)$ gives the precision at recall *r*. *AP* applies exactly 11 equally spaced recall levels: R_{11} = {0, 0.1, 0.2, ...,1} and $AP|_{R40}$ applies recall levels: R_{40} = {1/40, 2/40, 3/40,..., 1}. We mainly use *AP* as an accuracy indicator and $AP|_{R40}$ will be applied in the ablation experiment in Section 4.5.

In the nuScenes dataset, as mentioned above, we apply the *NDS* and *mAP* as the evaluation indicator. The specific formulas for *NDS* are expressed as follows:

$$NDS = \frac{1}{10}\left[5mAP + \sum_{mTP\in TP}(1 - \min(1, mTP))\right] \tag{10}$$

where *mTP* is the mean True Positive metrics and consists of 5 metrics: average translation error, average scale error, average orientation error, average velocity error, and average attribute error.

4.2. Experimental Setting

SAE3D is implemented based on the Appended [25] and is trained on a single GPU. All experiments were performed on Ubuntu 16.04 and NVIDIA RTX-2080Ti.

4.2.1. Setting in KITTI Dataset

During the training process, the batch size takes the value of 2, and 16,384 points are randomly selected from the remaining points in each batch to input into the detector. In terms of network parameters, the number of key points in the three SA layers is set to 4096, 1024, and 512, respectively, and the scaling factor Scale for the distance feature takes the value of 120.

Adam optimizer [26] and a periodically varying learning rate were adopted in the training for a total 80 epochs, with the initial value of the learning rate set to 0.001. Additionally, we used three commonly used data augmentation methods during training: randomly flipping the X-axis with respect to the Y-axis, random scaling, and randomly rotating the Z-axis.

4.2.2. Setting in nuScenes Dataset

During the training process, the batch size takes the value of 1. Adam optimizer and a periodically varying learning rate were adopted in the training for a total of 10 epochs, with the initial value of the learning rate set to 0.001.

To handle the huge number of points in the nuScenes dataset, four SA layers are adopted. The number of key points in these four SA layers is set to 16,384, 4096, 3072, and 2048, respectively.

4.3. Results

The detection performance of the SAE3D model is evaluated on the KITTI dataset and nuScenes dataset against some existing methods proposed in the literature.

In the KITTI dataset, the test set is categorized into three levels of difficulty, i.e., "Easy", "Moderate", and "Hard", based on the difficulty of detection. We take the 3D bounding box average precision (3D AP) of the "Car" category as the main evaluation, as this is usually adopted as the main indicator in KITTI datasets. As shown in Table 1, compared with the baseline network SASA, 3D AP is improved by 0.544% and 0.648% in the difficulty levels of

"Moderate" and "Hard", respectively. The detailed precision improvements will be shown in Section 4.5.

In the nuScenes dataset, as shown in Table 2, compared with the baseline network, SAE3D achieved 3.3% and 1.7% improvement in the indicators of NDS and mAP, respectively.

Table 1. The detection results of 3D AP for "Car" in KITTI.

Methods		**Car 3D AP (%)**	
	Easy	**Moderate**	**Hard**
SECOND [8]	84.656	75.966	68.712
Voxel Net [7]	77.478	65.119	57.736
Point Pillars [14]	82.588	74.317	68.995
Point-RCNN [18]	89.023	78.246	77.554
Vox Set Tran [10]	88.869	78.766	77.576
SASA [3]	**89.108**	78.847	77.588
SAE3D	89.059	**79.391**	**78.236**

Table 2. Results from the nuScenes validation set. Evaluation metrics include NDS, mAP, and 10 classes. Abbreviations: pedestrian (PED.), traffic cone (T.C.), construction vehicle (C.V.).

Methods	**NDS**	**mAP**	**Car**	**Truck**	**Bus**	**Trailer**	**C.V.**	**Ped.**	**Motor**	**Bicycle**	**T.C.**	**Barrier**
Point Pillars [14]	45.2	25.8	70.3	32.9	44.9	18.5	4.2	46.8	14.8	0.6	7.5	21.3
3DSSD [2]	51.7	34.5	75.9	34.7	60.7	21.4	10.6	59.2	25.5	7.4	14.8	25.5
SASA [3]	55.3	36.1	71.7	42.2	63.5	29.6	12.5	62.6	27.5	9.1	12.2	30.4
SAE3D	**58.6**	**37.8**	72.4	44.1	62.7	31.2	15.9	60.4	30.1	12.8	10.1	31.6

4.4. Enhancement Validation

To verify the enhancement effect of the proposed network in this paper, we utilize SASA [3] and Point-RCNN [18] as two baseline networks for testing in the KITTI dataset. Both baseline networks are point-based 3D object detection networks, where SASA [3] is a one-stage object detection network and Point-RCNN [18] is a two-stage object detection network. The experiments introduce the enhanced network proposed in this paper into both of these networks, effectively improving the detection performance of the original benchmark network.

Table 3 shows the improvement in the accuracy of the 3D detection frames of the "Car" category in the enhanced networks of SASA [3] and Point-RCNN [18], respectively.

Table 3. Enhancement effectiveness. Abbreviations: Distance features-based enhancement network proposed in this paper (SAE3D).

Methods		**Car 3D AP (%)**	
	Easy	**Moderate**	**Hard**
SASA [3]	89.108	78.847	77.588
SASA [3] **+ SAE3D**	89.059	79.391	78.236
Improvement	−0.049	+0.544	+0.648
Point-RCNN [18]	89.023	78.246	77.554
Point-RCNN [18] **+ SAE3D**	89.160	78.839	78.439
Improvement	+0.137	+0.593	+0.885

After the introduction of the enhanced network in SASA [3], the 3D AP of the "Car" decreases slightly in the "Easy" difficulty, but increases by 0.544% and 0.648% in the "Moderate" and "Hard" difficulties, respectively.

After introducing the enhanced network in Point-RCNN [18], the accuracy of the 3D AP is improved by 0.137%, 0.593%, and 0.885% in "Easy", "Moderate", and "Hard" difficulties, respectively.

4.5. Ablation Experiment

In this paper, ablation experiments are designed to verify the actual effect of each module. All modules are trained on the training set of the KITTI dataset and evaluated on the validation set for the "Car" category of the KITTI dataset.

In this section we added BBox AP, BEV AP, and AOS AP alongside 3D AP as the evaluation indicator. BBox AP represents the average precision of the 2D bounding box, while BEV AP denotes the average precision of the detection boxes in bird's-eye view. These two indicators provide detection box accuracy from different perspectives, aiding in a better understanding of the spatial precision of the detection boxes predicted by our model. AOS AP stands for the average precision of the detected target's rotation angle, indicating the accuracy of the object orientation predicted by our model.

4.5.1. Initial Feature Fusion Module

As shown in Table 4, the initial feature fusion module proposed in this paper is of great help to improve the precision of 3D bounding box. The improvement of this module is most evident in the difficulty levels of "Moderate" and "Hard". Compared to the baseline network used in this paper, in the "Moderate" and "Hard" difficulty levels, the 3D bounding box accuracy improvement of this module is 0.551% and 0.811%, respectively. Additionally, the improvement in 2D bounding box accuracy is 0.186% and 0.811%, while the bounding box accuracy improvement in BEV view is 0.257% and 1.048%, respectively.

Table 4. Comparison table of the general accuracy enhancement effect of different modules. Abbreviations: initial feature fusion module (I), KSFE module (K), and RGA module (F).

+I	+K	+F	Car 3D AP (%)			Car BBOX AP (%)			Car BEV AP (%)			Car AOS AP (%)		
			Easy	Mod	Hard	Easy	Mod	Hard	Easy	Mod	Hard	Easy	Mod	Hard
			89.108	78.847	77.588	96.742	89.855	89.036	90.199	87.855	85.993	96.71	89.75	88.88
	√		88.971	79.246	78.334	96.473	89.847	89.163	**90.317**	**88.420**	**87.342**	96.44	89.81	89.07
		√	89.213	79.324	78.114	**96.813**	**90.171**	**89.412**	89.876	89.397	86.976	96.54	90.08	89.11
√			**89.167**	**79.398**	**78.399**	96.668	90.041	89.287	90.149	88.112	87.041	96.64	89.98	89.12
√	√	√	89.059	79.391	78.236	96.758	90.169	89.382	89.978	88.382	86.824	**96.71**	**90.10**	**89.22**

As shown in Table 5, when using the AP$|_{R40}$, the improvement in the accuracy of the 3D bounding box is 2.549% and 2.582% for the difficulties of "Moderate" and "Hard", respectively. The improvement in the accuracy of 2D bounding box is 1.976% and 0.533%, respectively, and the improvement in the accuracy of bounding box in BEV view is 0.295% and 2.257%, respectively.

Table 5. Comparison table of the AP$|_{R40}$ enhancement effect of different modules. Abbreviations: initial feature fusion module (I), KSFE module (K), and RGA module (F).

+I	+K	+F	Car 3D AP R40 (%)			Car BBOX AP R40(%)			Car BEV AP R40(%)			Car AOS AP R40(%)		
			Easy	Mod	Hard	Easy	Mod	Hard	Easy	Mod	Hard	Easy	Mod	Hard
			91.592	80.705	77.902	**98.289**	92.972	92.104	93.277	89.128	86.465	**98.26**	92.85	91.92
	√		91.457	83.067	80.369	98.111	94.583	92.469	**95.055**	**91.034**	**88.832**	98.09	94.52	92.35
		√	91.432	82.913	78.956	98.023	95.023	92.659	93.124	89.223	88.624	98.21	94.89	92.39
√			91.555	**83.254**	**80.484**	98.097	94.948	**92.637**	93.197	89.423	88.722	98.08	94.85	**92.44**
√	√	√	91.426	83.236	80.191	98.266	**95.036**	92.616	93.014	90.902	88.525	98.23	**94.93**	92.42

4.5.2. Key Points Self-Features Enhancement Module

As shown in Table 4, this module improves the detection accuracy of the 3D bounding box and the accuracy of bounding box detection in BEV view. The detection accuracy of the 3D bounding box is improved by 0.339% and 0.746% under the difficulty levels of "Moderate" and "Hard", respectively, and the detection accuracy of the bounding box in BEV view is improved by 0.118%, 0.565%, and 1.349% in "Easy", "Moderate", and "Hard" levels of difficulty, respectively.

As shown in Table 5, the accuracy of the 3D bounding box is improved by 2.362% and 2.467% for the "Moderate" and "Hard" levels of difficulty, respectively, when using AP $|_{R40}$. The accuracy of the bounding box in BEV view is improved by 1.778%, 1.906% and 2.367% for "Easy", "Moderate", and "Hard" levels of difficulty, respectively.

4.5.3. Revised Group Aggregation Module

As shown in Table 4, the detection accuracy of this module on BBOX is improved by 0.316% and 0.376% under the difficulty levels of "Moderate" and "Hard", respectively. Additionally, compared with the baseline network, the module improves other metrics such as 3D bounding box and steering angle accuracies.

As shown in Table 5, when AP $|_{R40}$ is used, the detection accuracy improvement on BBOX is 2.051% and 0.555% at "Moderate" and "Hard" levels, respectively.

4.6. Detection Effect

Figure 6 shows the actual detection effect. Although there is still a small part of the missed detection problem, most of the vehicles are detected and the accuracy of the 3D bounding box is high.

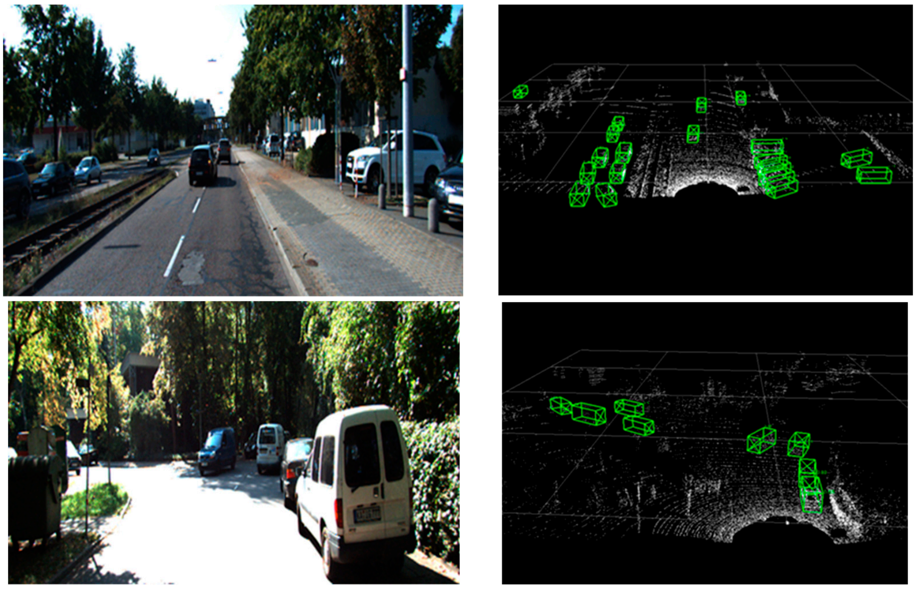


Figure 6. Actual detection effect diagram in KITTI dataset (left are the pictures of the real scenes, right are the detection 3D bounding boxes predicted in the point cloud).

5. Discussion

In this paper, we continue to explore the possibilities of the point-based 3D object detection. Point cloud data are vast and contains a wealth of information, both useful and redundant. We believe that there is still underutilized information within the point cloud. Therefore, we proposed the SAE3D. The results demonstrate that extracting more useful information and enhancing the relevant information in the point cloud can improve the final detection accuracy.

6. Conclusions

In this paper, we proposed SAE3D with three enhancement modules: an initial feature fusion module, a key points self-feature enhancement module, and a revised group aggregation module. We provide a detailed description of the design ideas and implementation of these modules in this paper. We conducted testing using the KITTI and nuScenes datasets and designed ablation experiments on the KITTI dataset to analyze the enhancement of each module in detail. The results demonstrate that all three enhancement modules we propose contribute to enhancing detection accuracy. Our SAE3D suggests that there are still useful characteristics in point clouds that are not fully utilized, and some of them can assist in extracting information from the point clouds more effectively. We believe that exploring additional potential characteristics of point clouds can further enhance 3D scene understanding.

Author Contributions: Conceptualization, Z.Z. and Z.B.; methodology, Z.Z. and Z.B.; software, Z.B.; validation, Z.Z.; formal analysis, Z.B.; investigation, Q.T. and Z.B.; resources, Q.T.; data curation, Z.B.; writing, Z.Z. and Z.B.; original draft preparation, Z.B.; visualization, Z.Z.; supervision, Z.B., Z.L. and Q.T.; project administration, Q.T. and Z.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key Research and Development Program under Ministry of Science and Technology of the People's Republic of China.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Adv. Neural Inf. Process. Syst.* **2017**, *30*.
2. Yang, Z.; Sun, Y.; Liu, S.; Jia, J. 3DSSD: Point-based 3d single stage object detector. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11040–11048.
3. Chen, C.; Chen, Z.; Zhang, J.; Tao, D. Sasa: Semantics-augmented set abstraction for point-based 3d object detection. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 22 February–1 March 2022; 36, pp. 221–229.
4. Sun, X.; Wang, S.; Wang, M.; Cheng, S.S.; Liu, M. An advanced LiDAR point cloud sequence coding scheme for autonomous driving. In Proceedings of the 28th ACM International Conference on Multimedia, Seattle, WA, USA, 12–16 October 2020; pp. 2793–2801.
5. Sun, X.; Wang, M.; Du, J.; Sun, Y.; Cheng, S.S.; Xie, W. A Task-Driven Scene-Aware LiDAR Point Cloud Coding Framework for Autonomous Vehicles. *IEEE Trans. Ind. Inform.* **2022**, *19*, 8731–8742. [CrossRef]
6. Huang, R.; Wang, M. Patch-Wise LiDAR Point Cloud Geometry Compression Based on Autoencoder. In Proceedings of the International Conference on Image and Graphics, Nanjing, China, 22–24 September 2023; Springer Nature: Cham, Switzerland, 2023; pp. 299–310.
7. Zhou, Y.; Tuzel, O. Voxelnet: End-to-end learning for point cloud based 3d object detection. In Proceedings of the IEEE con-Ference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4490–4499.
8. Yan, Y.; Mao, Y.; Li, B. Second: Sparsely embedded convolutional detection. *Sensors* **2018**, *18*, 3337. [CrossRef] [PubMed]
9. Mao, J.; Xue, Y.; Niu, M.; Bai, H.; Feng, J.; Liang, X.; Xu, H.; Xu, C. Voxel transformer for 3d object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 3164–3173.
10. He, C.; Li, R.; Li, S.; Zhang, L. Voxel set transformer: A set-to-set approach to 3d object detection from point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 8417–8427.
11. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*.
12. Lee, J.; Lee, Y.; Kim, J.; Kosiorek, A.; Choi, S.; Teh, Y.W. Set transformer: A framework for attention-based permutation-invariant neural networks. In Proceedings of the International Conference on Machine Learning PMLR, Long Beach, CA, USA, 10–15 June 2019; pp. 3744–3753.
13. Zhu, L.; Chen, Z.; Wang, B.; Tian, G.; Ji, L. SFSS-Net: Shape-awared filter and sematic-ranked sampler for voxel-based 3D object detection. *Neural Comput. Appl.* **2023**, *35*, 13417–13431. [CrossRef]

14. Lang, A.H.; Vora, S.; Caesar, H.; Zhou, L.; Yang, J.; Beijbom, O. Pointpillars: Fast encoders for object detection from point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 12697–12705.
15. Shi, G.; Li, R.; Ma, C. Pillarnet: High-performance pillar-based 3d object detection. *arXiv* **2022**, arXiv:2205.07403.
16. Li, J.; Luo, C.; Yang, X. PillarNeXt: Rethinking network designs for 3D object detection in LiDAR point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 18–22 June 2023; pp. 17567–17576.
17. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660.
18. Shi, S.; Wang, X.; Li, H. Pointrcnn: 3d object proposal generation and detection from point cloud. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 770–779.
19. Chen, Y.; Xu, F.; Chen, G.; Liang, Z.; Li, J. Point cloud 3D object detection method based on density information-local feature fusion. *Multimed. Tools Appl.* **2023**, 1–19. [CrossRef]
20. Ding, Z.; Han, X.; Niethammer, M. Votenet: A deep learning label fusion method for multi-atlas segmentation. In *Medical Image Computing and Computer Assisted Intervention, Proceedings of the MICCAI 2019: 22nd International Conference, Shenzhen, China, 13–17 October 2019*; Part III 22; Springer International Publishing: Berlin/Heidelberg, Germany, 2019; pp. 202–210.
21. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The kitti dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [CrossRef]
22. Caesar, H.; Bankiti, V.; Lang, A.H.; Vora, S.; Liong, V.E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; Beijbom, O. nuscenes: A multimodal dataset for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11621–11631.
23. Salton, G.; McGill, M.J. *Introduction to Modern Information Retrieval*; McGraw-Hill, Inc.: New York, NY, USA, 1986.
24. Simonelli, A.; Bulo, S.R.; Porzi, L.; Lopez-Antequera, M.; Kontschieder, P. Disentangling monocular 3d object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1991–1999.
25. OD Team. Openpcdet: An Open-Source Toolbox for 3d Object Detection from Point Clouds. *OD Team* **2020**.
26. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

sensors

MDPI

Article

RRGA-Net: Robust Point Cloud Registration Based on Graph Convolutional Attention

Jian Qian [†] and Dewen Tang [*,†]

School of Mechanical Engineering, University of South China, Hengyang 421001, China; 20212006210312@stu.usc.edu.cn

[*] Correspondence: will998@126.com

[†] These authors contributed equally to this work.

Abstract: The problem of registering point clouds in scenarios with low overlap is explored in this study. Previous methodologies depended on having a sufficient number of repeatable keypoints to extract correspondences, making them less effective in partially overlapping environments. In this paper, a novel learning network is proposed to optimize correspondences in sparse keypoints. Firstly, a multi-layer channel sampling mechanism is suggested to enhance the information in point clouds, and keypoints were filtered and fused at multi-layer resolutions to form patches through feature weight filtering. Moreover, a template matching module is devised, comprising a self-attention mapping convolutional neural network and a cross-attention network. This module aims to match contextual features and refine the correspondence in overlapping areas of patches, ultimately enhancing correspondence accuracy. Experimental results demonstrate the robustness of our model across various datasets, including ModelNet40, 3DMatch, 3DLoMatch, and KITTI. Notably, our method excels in low-overlap scenarios, showcasing superior performance.

Keywords: point cloud registration; deep learning; attention mechanism

1. Introduction

Point cloud registration is a crucial task within computer vision and robotics, frequently applied in significant domains like autonomous driving [1], 3D reconstruction [2], and simultaneous localization and mapping (SLAM) [3]. In recent years, with the development of point cloud processing technology and deep learning technology, point cloud coding algorithms such as [4,5] have further optimized the processing of large-scale point cloud data and improved the accuracy of point cloud registration in large-scale scenes. However, registration in low-overlap environments remains a challenging task.

A classical point cloud registration method is Iterative Nearest Point (ICP) [6]. It starts with an initial transform guess, and then iteratively updates the transform matrix to minimize the distance between corresponding points until convergence or a certain number of iterations is reached. The disadvantage is that it is sensitive to initial transformations and local minima. The Fast Global Registration (FGR) [7] algorithm addresses the drawbacks of the ICP algorithm through global alignment, but is still prone to failure in noisy environments.

Deep learning-based methods can be divided into three categories: the first category [8,9] is based on the global features of the point cloud, which is treated as a whole to regress the transformation parameters. Although this type of method has good robustness to noise, it is not effective for the registration task of partially overlapping point cloud. The second class of methods [10,11] based on correspondence learning form correspondences by means of high-dimensional features of the points and iteratively minimize the feature distances to optimize the pose. This type of approach extracts the correspondences of the points, so it is more robust in partially overlapping point cloud registration, but it is still susceptible to noise interference. The last class [12–14] uses a two-stage approach to the point cloud correspondences, where they first learn the local descriptors of the downsampled keypoints for matching, and then use a pose estimator to recover the relative transformations. Their strategy allows them to achieve state-of-the-art performance in the dataset, but the downsampling method they use inherently introduces sparsity point cloud features, which reduces the reproducibility of the correspondences and thus loses its advantage in low overlap regions.

A recently discovered approach [15] bypasses direct keypoint detection. It mitigates the limitations of keypoint detection by employing a coarse-to-fine strategy similar to the two-dimensional correspondence approach [16,17] to address the problem of keypoint sparsity. It achieves correspondence extraction by employing a superpoint patch-to-point path. Essentially, the method extracts superpoint features from the original point cloud, merges them into superpoint patches for matching, and then extends the matched correspondences to include dense points. The advantage of the method is that it transforms the strict point matching requirement into a more relaxed patch overlapping environment. This shift effectively reduces the requirement for a large number of repeatable keypoints. However, it also emphasizes the importance of keypoint reliability. Furthermore, while this approach reduces the need for a large number of keypoints, its sparsity remains unchanged. Therefore, in this paper, more emphasis is placed on compensating for this inherent sparsity by capturing contextual features.

Taking inspiration from references [15,18], our initial focus lies in optimizing the enhancement of the patch context function. With the objective in mind, a matching module based on a graph convolutional neural network was designed, the core of which is constituted by two modules: the self-attention graph convolutional neural network and the cross-attention module. Graph convolutional networks focus on different points when processing point cloud data and dynamically adjust weights based on the relationship between them. This adaptability enables the model to better capture local and global features in point cloud, thereby improving the modeling ability of template points. Secondly, the cross-attention module enables the model to effectively capture the information interaction in point cloud between different channels by introducing cross-channel correlation. This helps integrate multi-channel information to more fully understand feature relationships in point cloud. Through this cross-channel interaction, the model is better able to adapt to complex structures in point cloud data. In addition, previous methods use grid-sampled superpoints as nodes and divide patches through a point-to-node grouping strategy. Since the grid-sampled superpoints are sparse and loose, the local neighborhoods between point cloud pairs are inconsistent, which adversely affects subsequent point matching. To this end, it is recommended to extract points characterized by a more uniform visual field distribution and high repeatability as nodes. The utilization of the feature pyramid effect is suggested for scoring nodes across various receptive fields. Additionally, incorporating multi-level point cloud sampling is advised during the process of patch fusion. During the sampling of point cloud at different levels, the density of sampled points is regulated through non-maximum suppression. Utilizing multi-channel sampling and point matching comparison allows for the acquisition of more comprehensive point cloud information, resulting in improved correspondence with template points.

2. Related Work

2.1. Traditional Point Cloud Registration Methods

The Iterative Closest Point (ICP) [6] algorithm has retained its practical importance since it became established. Its straightforward logic and ease of implementation have solidified its position as a staple method for aligning rigid-body transformations. It shows strong convergence when the initial deviation is relatively small. However, its accuracy decreases when the initial bias is large, resulting in local optima and sensitivity to noise. Consequently, an array of ICP-based variants [19–21] have emerged.These ICP-based variations expand upon the original concept to address its limitations. They offer enhanced flexibility by accommodating multiple constraints such as distance, geometry, and normals, leading to improved robustness against initial deviations. However, this enhancement often comes at the cost of increased computational complexity.

Among alternative strategies, feature-based registration methods like Fast Point Feature Histograms (FPFH) [22] and Signature of Histograms of Orientations (SHOT) [23] have been developed to adapt to complex environments and noise by extracting local features. The Random Sample Consensus (RANSAC) [24] divides the point cloud into random subsets for local registration, effectively eliminating outliers, albeit at the expense of computational time. Conversely, the Fast Global Registration (FGR) [7] transforms the non-convex problem into a convex one through a smoothing mechanism, enabling rapid global registration. However, this method exhibits heightened sensitivity to errors.

2.2. Deep Learning-Based Methods

PointNet [25] is the first deep learning model that can directly process raw point cloud data without converting the point cloud into voxels or meshes. Thus, PointNetLK [8] uses PointNet's ability to extract global features from point cloud to achieve alignment via the LK optical flow method in classical image alignment. PCRNet [9], on the other hand, uses a multilayer perceptron (MLP) to solve the transformation parameters by treating the registration task as a regression problem after extracting features using PointNet. But PointNet, although it is easy to extract global features of the point cloud, loses the value of local features, so it is not robust to noisy and partially overlapping environments.

In contrast to approaches relying on PointNet, DCP [10] uses a Dynamic Graph Convolutional Neural Network (DGCNN) [26] to extract local features from the original point cloud. The rotation matrix and translation parameters are then computed by Singular Value Decomposition (SVD). RPMNet [27] introduces an auxiliary network to predict the optimal asymptotic annealing parameters to derive soft matches for point correspondences in integrating spatial coordinates and local geometric features. These approaches based on local correspondence learning solve part of the point cloud matching problem to some extent, but the network fails to converge when the rotation angle is too large. In the method of correlated feature point detection, D3Feat [14] utilizes a fully convolutional network architecture for joint dense detection and description. Predator [12] predicts dense overlap scores on top of jointly estimating significant scores and learning local descriptors, and analyses the confidence level of whether a point is located on an overlapping region. However, they show low robustness in low overlap scenarios. This is due to the fact that they still inherently rely on repeatable keypoints.

A facet-to-point correspondence strategy is employed, and the establishment of point cloud correspondence is carried out in a coarse-to-fine manner. The dependence on repeatable key points is reduced by composing patches. Meanwhile, the multi-channel sampling strategy has a denser set of correspondence points, while the attention mechanism-based graph convolutional neural network enriches the correspondence of template points by interacting with contextual features. The performance of various algorithms will be compared in Section 4.

3. Methods

Point cloud registration involves aligning point cloud data of an object from distinct viewpoints or sensors onto the same coordinate system. The objective is to fit multi-frame images, enhance visual perception to facilitate understanding of the environment, and provide assistance in subsequent tasks.

The central challenge of point cloud registration revolves around aligning two distinct point clouds, $\mathcal{P} = \{\mathbf{p}_i \in \mathbb{R}^3 \mid i = 1, ..., N\}$ and $\mathcal{Q} = \{\mathbf{q}_i \in \mathbb{R}^3 \mid i = 1, ..., M\}$, while optimizing their alignment within the shared coordinate system. This task can be represented through the subsequent model: how can we determine a transformation matrix, denoted as $\mathbf{T} = \{\mathbf{R}, \mathbf{t}\}$, which effectively reposition point cloud $\mathcal{Q}$ to attain optimal conformity with the spatial orientation of point cloud $\mathcal{P}$? This issue can be effectively characterized as an optimization problem:

$$\arg\min_{\mathbf{R},\mathbf{t}} \sum_{i=1}^{N} \sum_{i=1}^{M} \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2 \tag{1}$$

where $\mathbf{R}$ denotes the rotation matrix and $\mathbf{t}$ denotes the translation parameter. We estimate the alignment transformation by finding point correspondences.

The point cloud registration model is illustrated in Figure 1. KPConv and FPN are utilized simultaneously to downsample the input point cloud and extract features (refer to Section 3.1). The downsampled points of the three different levels of the three layers are also selected as the reference and feature points of the correspondences to be matched, respectively. The graph convolutional neural matching module is used to extract the correspondences of the feature points (Section 3.2). Subsequently, the point matching module employs these correspondences to extend the alignment from feature points to encompass the entire dense point cloud (Section 3.3). At last, the local-to-global alignment method estimates the transformation matrix.

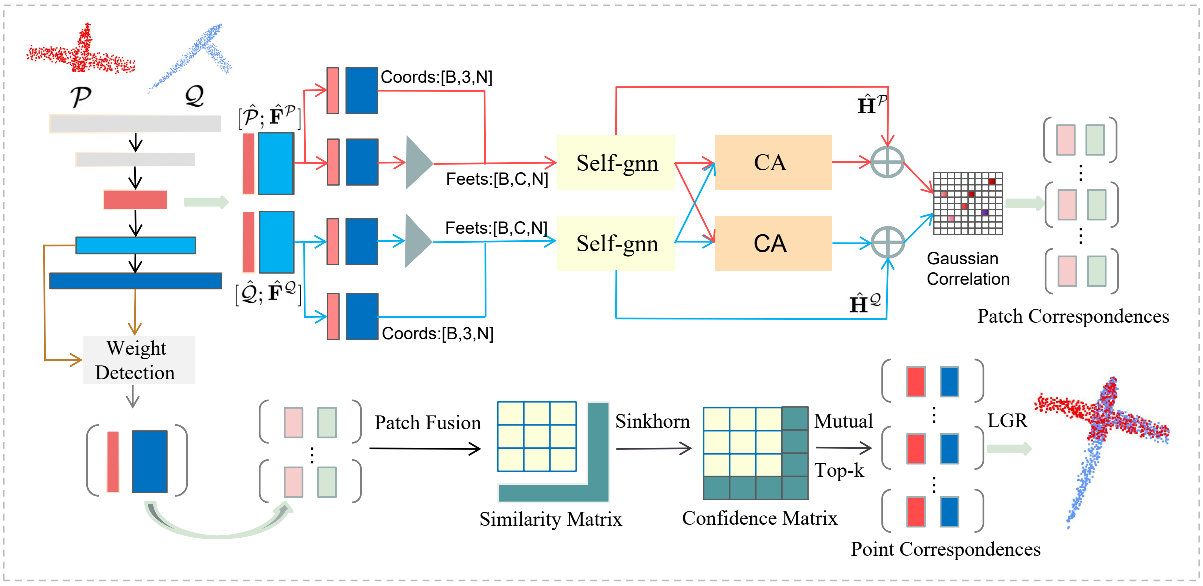


Figure 1. Given two partially overlapping point clouds $\mathcal{P}$ and $\mathcal{Q}$, we first adopt KPConv-FPN to learn point cloud features at different levels. $\bar{\mathbf{F}}^{\mathcal{P}}$ or $\bar{\mathbf{F}}^{\mathcal{Q}}$ are connected to the centroid features through the K-NN algorithm and then imported into the graph convolutional neural network matching module. Their contextual features are further aggregated and enhanced through the self-attention graph convolution network (Self-gnn) and cross-attention (CA) block. Subsequently, the patch correspondences is mapped to the real point set through the point matching module. Finally, a local-to-global registration method is used to calculate the transformation matrix.

3.1. Feature Extraction

The KPConv-FPN [28] technique is utilized to downsample the initial point cloud and obtain features at the individual point level. Multiple resolution levels are sampled from the original point cloud $\mathcal{P}$. These layers of sampled point cloud exhibit progressively diminished resolution connections, necessitating a coarse-to-fine methodology to master correspondences. The points corresponding to the most rudimentary resolution, designated as $\hat{\mathcal{P}}$, are regarded as the reference points to be aligned. Both multi-level transition points, denoted as $\bar{\mathcal{P}}$, and dense points, denoted as $\tilde{\mathcal{P}}$, are independently extracted, and their respective acquired features are labeled as $\bar{\mathbf{F}}^{\mathcal{P}} \in \mathbb{R}^{|\bar{\mathcal{P}}| \times \bar{d}}$ and $\tilde{\mathbf{F}}^{\mathcal{P}} \in \mathbb{R}^{|\tilde{\mathcal{P}}| \times \tilde{d}}$. d is the corresponding sampling scale, determined by the resolution factor. For each modal point, a point-to-node grouping strategy is employed, thereby constructing localized point patches around it. The features in $\bar{\mathbf{F}}$ and $\tilde{\mathbf{F}}$ are assigned to the nearest modal point, and the ensuing equation validates the assigned weights:

$$\mathbf{w} = \|\tilde{\mathbf{p}} - \hat{\mathbf{p}}\|_2 / \|\bar{\mathbf{p}} - \hat{\mathbf{p}}\|_2 \tag{2}$$

Based on the weights, for different levels, the nearest points will be attributed to the modal points closest to them and the resulting patch is shown below:

$$\mathcal{G}_i^{\mathcal{P}} = \begin{cases} i = \text{argmin}_j(\|\bar{\mathbf{p}} - \hat{\mathbf{p}}_j\|_2), \hat{\mathbf{p}}_j \in \hat{\mathcal{P}} & \mathbf{w} > 1, \\ i = \text{argmin}_j(\|\tilde{\mathbf{p}} - \hat{\mathbf{p}}_j\|_2), \hat{\mathbf{p}}_j \in \hat{\mathcal{P}} & \mathbf{w} \leq 1, \end{cases} \tag{3}$$

3.2. Graph Convolutional Neural Network Matching Module

Obtaining a global field of view is critical in a variety of computer vision tasks. Therefore, we employ an attention mechanism that utilizes broader contextual information to augment global properties. This yields enhanced geometric differentiation within the acquired features, consequently mitigating pronounced matching ambiguities and a surplus of aberrant matches, particularly in scenarios characterized by limited overlap. Through the utilization of the cross-focusing mechanism, feature details from the point cloud can be adeptly interchanged and amalgamated, leading to the identification of pivotal points connected with regions of overlap. This innovative approach effectively solves the problem of redundant point accumulation while simplifying the process of selecting point sets during the alignment process. In the cross-focus module, the initial embedding contains features from both the source and target point cloud.

Before connecting the feature codes of the two inputs, a graph neural network (GNN) is first used to further aggregate and strengthen their contextual relationships, respectively. The point sets $\mathcal{P}$ or $\mathcal{Q}$ are connected into a graph within the Euclidean space through the employment of the K-Nearest Neighbors (K-NN) algorithm. Subsequently, utilizing K-NN searches based on coordinates, the features are linked to centroid features.

$$f_i = cat(x_i^n, x_j^n - x_i^n) \tag{4}$$

In the aforementioned equation, x^n signifies the feature encoding corresponding to the point set $\mathcal{P}$, while "cat" denotes concatenation.

$$h_i = \text{LeakyRLU}(\text{norm}(\text{conv}(f_i))) \tag{5}$$

$$\mathrm{x}_i^{n+1} = \max_{(i,j)\in\varepsilon} h_i(f_i) \tag{6}$$

where $\mathbf{h_i}$ denotes the linear layer, norm denotes the activation function, max denotes the elemental channel maximum pooling layer, and $(i, j) \in \varepsilon$ denotes the two edges of the graph.

$$\mathbf{x}_i^{self-GNN} = h_i(cat(x_i^0, x_i^1, x_i^2)) \tag{7}$$

Cross-attention stands as a prototypical module within point cloud registration tasks, fostering the exchange of features between two input point clouds. We apply self-attention processing to the three-channel point cloud data, combining the resultant data with convolved point cloud information, and subsequently activate the aggregated point cloud using a Multi-Layer Perceptron (MLP). The computation of the Cross-Attention (CA) module is detailed as follows:

$$m_i = \text{att}(x_i, x_j, x_j) \tag{8}$$

Here, att denotes Multi Head Attention. $x_i = x_i^{self-GNN}$,$x_j = x_j^{self-GNN}$.

$$\mathbf{X}_i^{CA} = x_i^{self-GNN} + \text{MLP}(\text{cat}(x_i^{self-GNN}, m_i)) \tag{9}$$

In the sampling of correspondences, considering patch correspondences at different levels helps to obtain more robust point correspondences in the point matching stage. However, due to the sparse and loose nature of block matching, many correct correspondences are often overlooked in the screening process. Our proposed multi-channel convolutional network can complement more effective point correspondences for accurate point cloud registration.

3.3. Point Matching Module

After obtaining the template point correspondence, the dense point correspondence will be derived based on the patch correspondence. Subsequently, the local-to-global registration (LGR) mechanism derives candidate matrices from the point correspondences engendered by each pairing of matching patches. From these candidates, the globally optimal transformation matrix is selected. Pertaining to the point-level, our approach exclusively employs localized point features gleaned from the backbone network. The underlying principle is that, upon resolving global ambiguity through template point matching, point-level matching is predominantly influenced by the proximity of the matched points. This strategic design enhances the overall robustness of the process.

For each template point correspondence, the optimal transport layer is employed to derive localized dense point correspondences between the point clouds. The process begins by calculating the cost matrix:

$$\mathbf{C}_i = \mathbf{F}^{\mathcal{P}}_{x_i}(\mathbf{F}^{\mathcal{Q}}_{y_i})^T / \sqrt{\tilde{d}}, \tag{10}$$

Following this, the cost matrix undergoes expansion through the addition of new rows and columns, infused with learnable bin parameters. Subsequently, the Sinkhorn algorithm is employed to compute the soft assignment matrix. This matrix is then reinstated to its original form by discarding the last row and column. This resultant matrix serves as a confidence measure for prospective matches. Point correspondences are subsequently culled through mutual top-k selection, whereby a point match is affirmed if it falls within the k largest entries within its respective row and column. The point correspondences computed from each template point match are then collected together to form the final global dense point correspondence: $\mathcal{C} = \bigcup_{i=1}^{N_c} \mathcal{C}_i$.

3.4. Loss Functions

The loss functions of the Graph Convolutional Neural Matching Module and the Point Matching Module are divided into the following two points.

A metric learning approach is chosen to cultivate a feature space that facilitates the assessment of the similarity of samples. This approach is tailored to more effectively evaluate the matching interrelation between patches, facilitating the convergence of matches and divergence of mismatches. We meticulously select patches within $\mathcal{P}$, ensuring each belongs to a group of anchor point patches denoted as $\mathcal{N}$, where a positive patch in $\mathcal{Q}$ is present. Pairs of patches are categorized as positive if they display a minimum of 10% overlap, and conversely as negative if they lack overlap. All other pairs are omitted from consideration. For each anchor patch $\mathcal{G}_i \in \mathcal{N}$, a corresponding loss takes on the following format:

$$\mathcal{L}_c = \sum_{\mathcal{G}_i^{\mathcal{P}} \in \mathcal{N}} \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+ \tag{11}$$

where x_i^a symbolizes the feature representation of the anchor, while x_i^p signifies the feature representation of the positive example that aligns with the anchor patch. Correspondingly, x_i^n stands for the feature representation of the negative example, which does not align with the anchor patch. The parameter α functions as a constant threshold, integral to guaranteeing that the disparity in distance between the positive and negative examples remains surpassing a predefined threshold. The function $[z]_+ = \max(z, 0)$ corresponds to the Rectified Linear Unit (ReLU) function. By cultivating a suitable feature space and employing these strategies, we optimize the discernment of matching relationships among patches, thereby enhancing the precision of point cloud registration.

The correspondence relationship of the real point set is sparser than that of the downsampled template points. The correspondence matrix $\mathbb{Z}$ in point matching is classified using a negative log-likelihood loss.

During training, true point correspondences $\hat{\mathcal{C}}i$ are randomly sampled. For each $\hat{\mathcal{C}}i$, a set of true point correspondences $\mathcal{M}$ is extracted using the matching radius r. The set of unmatched points in the two patches is denoted as $\mathcal{I}_i$ and $\mathcal{J}_i$. The individual point matching loss of $\hat{\mathcal{C}}_i$ is computed as:

$$\mathcal{L}_p = -\sum_{(x,y)\in\mathcal{M}} \log \bar{z}^i_{x,y} - \sum_{x\in\mathcal{I}} \log \bar{z}^i_{x,m_i+1} - \sum_{y\in\mathcal{J}} \log \bar{z}^i_{n_i+1,y} \tag{12}$$

The final loss function X consists of three loss functions together: $\mathcal{L} = \mathcal{L}_c + \mathcal{L}_m + \mathcal{L}_p$, where $\mathcal{L}_m$ is different from $\mathcal{L}_p$. The intermediate layer $\bar{\mathcal{P}}$ is used as the real point set. This approach establishes a link between multiple levels of point correspondences, and by exploiting these multiple levels of point correspondences, our method can compute point cloud feature parameters from a comprehensive perspective. This strategy not only improves the accuracy of point cloud alignment, but also enriches the representation learning process by exploiting the hierarchical structure inherent in the data.

4. Results

This section is dedicated to the experimental validation and performance comparison of our proposed method. The efficacy of the model is meticulously assessed through comprehensive experimentation. To establish a robust basis for evaluation, comparisons are conducted against several established methods, namely ICP, FGR, PointNetLK, DCP, and RPMNet. For the evaluation process, the ModelNet40 dataset [29] is employed as a benchmark. Through testing and analysis, distinctive advantages offered by our model in contrast to these existing methodologies are elucidated. Furthermore, to ascertain the generalizability of our approach in real-world scenarios, the evaluation is extended to encompass actual data. Engagement with the 3DMatch [30], 3DLoMatch [31], and KITTI [32] datasets is carried out to test the adaptability and reliability of our model within practical contexts.

4.1. ModelNet40

Our algorithm undergoes a thorough evaluation process on the ModelNet40 dataset, which encompasses computer-aided design (CAD) models representing 40 diverse classes of human-made objects. The evaluation strategy involves training on a set of 9756 models and testing on a separate collection of 2555 models. In alignment with the experimental framework established by RPMNet, adherence to specific guidelines is maintained. For each given shape, 1024 points are selected to constitute a point cloud. Additionally, an element of randomness is introduced into the evaluation process. Specifically, three Euler angles per point cloud are generated, each within the range of [0, 90°]. Furthermore, translations are introduced within the range of $[-0.5, 0.5]$. The original and target point clouds are distinguished in red and green.

A consistent metric framework is adopted, aligning with the assessment criteria employed by RPMNet [11] to evaluate the performance of our algorithm. This approach ensures comparability with previous research and underscores the reliability of our results. In this metric framework, the evaluation of alignment is performed by calculating the average isotropic rotation and translation errors, along with the average absolute errors of the Euler angles and translation vectors. If the overlapping regions of the two point clouds are identical, then all error parameters should be close to zero.

The performance of the algorithm is thoroughly evaluated across various point cloud scenarios, including clean point cloud, environments with noise, and instances of partially visible point cloud. The experimental outcomes are graphically presented in Figure 2 and

Tables 1–3. Since some algorithms do not have reliable open source implementations, some data come from their papers.

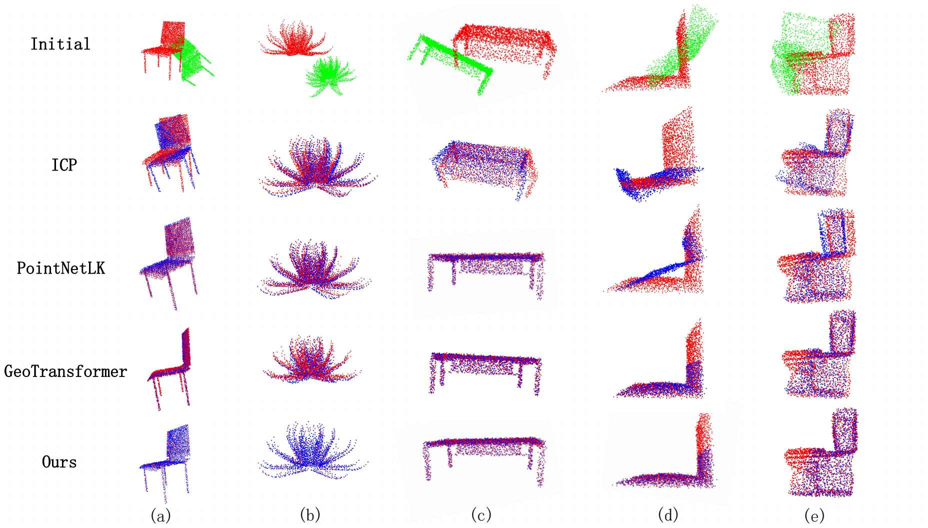


Figure 2. Examples of qualitative registration: (**a**,**b**) clean data, (**c**) noisy data, (**d**,**e**) partially visible noisy data.

Table 1. Performance on clean data.

Model	Isotropic R(°)	Isotropic t(m)	Anisotropic R(°)	Anisotropic t(m)	Time (s)
ICP	5.478	0.0765	11.443	0.1625	0.013
FGR	0.010	0.0001	0.022	0.0002	0.086
PointNetLK	0.418	0.0241	0.847	0.0054	0.157
DCP	2.074	0.0143	3.992	0.0292	0.009
PCR	2.691	0.0346	5.682	0.0735	0.059
GeoTransformer	0.072	0.0025	0.091	0.0023	0.023
Ours	0.034	0.0003	0.074	0.0005	0.052

Table 2. Performance on data with Gaussian noise.

Model	Isotropic R(°)	Isotropic t(m)	Anisotropic R(°)	Anisotropic t(m)	Time (s)
ICP	5.863	0.0823	12.145	0.1726	0.024
FGR	2.483	0.0325	4.274	0.0631	0.118
PointNetLK	1.528	0.0128	2.926	0.0262	0.214
DCP	4.528	0.0345	8.922	0.0707	0.020
PCR	2.943	0.0417	6.255	0.0804	0.122
GeoTransformer	1.156	0.0097	1.437	0.0213	0.045
Ours	0.525	0.0072	1.325	0.0127	0.083

Table 3. Performance on partially visible data with noise.

Model	Isotropic R(°)	Isotropic t(m)	Anisotropic R(°)	Anisotropic t(m)	Time (s)
ICP	13.719	0.132	27.250	0.280	0.017
FGR	19.266	0.090	30.834	0.192	0.124
PointNetLK	15.931	0.142	29.725	0.297	0.176
DCP	6.380	0.083	12.607	0.169	0.014
PCR	4.437	0.065	9.218	0.135	0.146
GeoTransformer	1.332	0.015	2.213	0.052	0.067
Ours	0.917	0.012	1.577	0.018	0.101

From Figure 2, it can be concluded that traditional methods such as ICP are susceptible to initialization, which is particularly obvious when the rotation angle is large. On the other hand, the efficacy of FGR is weakened in noisy environments because FPFH is sensitive to the noise problem under different point cloud conditions. In contrast, PointNetLK performs well in noisy environments, but still faces challenges in partially visible data. The reason for this phenomenon is that global feature methods emphasize the overall features of the point cloud rather than the specific local features of individual points. GeoTransformer works well in clean and noisy point clouds. Even in cases involving partially visible noise, superior registration results are observed for point clouds with simple structures (d). However, in the case of point cloud (e), our algorithm outperforms GeoTransformer. This is because injecting geometric information can improve performance, but the estimation method based on the geometric transformer does not rely on a stable estimator like RANSAC, which increases the difficulty in the estimation of actual super points, and GNN contains the transformation of the kNN graph, invariance, and better performance in transformation estimation. Therefore, our method improves the registration effect more significantly.

The tabular data in Tables 1–3 is further analyzed. Table 1 shows that FGR performs better than us in clean data, and then its performance in noisy data reflects the previous inference. The following focuses on comparing some visible noise point cloud data in Table 3. The PointNetLK algorithm, which performs well in Table 2, meets its Waterloo, and the DCP also suffers more in the absence of point clouds. Our algorithm still maintains a relatively excellent score, while still improving compared to the data of GeoTransformer. In terms of registration efficiency, ICP boasts the simplest algorithm structure. Given the small point cloud base in this experiment, it highlights the superiority of geometric algorithms. Neither FGR nor ICP utilizes the GPU, resulting in FGR's efficiency not being significantly enhanced after adding normal vector calculations. DCP adopts an end-to-end design model, eliminating the disadvantages of multi-stage calculation iterations seen in other algorithms, and it performs exceptionally well in computational efficiency due to GPU utilization. However, as the size of the point cloud increases, the performance of the end-to-end algorithm will experience nonlinear decline. GeoTransformer, utilizing geometric information to improve registration speed, achieves faster transformation estimation by omitting RANSAC. In contrast, our method utilizes multi-level feature extraction which, although it reduces part of the registration speed, increases the accuracy of feature extraction. Furthermore, the introduction of template points through hybrid sampling enhances the effectiveness of plane segmentation and feature matching.

4.2. Indoor Benchmarks: 3DMatch and 3DLoMatch

The point cloud data of the real environment is more complex than ModelNet40. The larger number of point cloud and lower overlap area will make many algorithms effective on ModelNet lose their advantages. The robustness of our algorithm is assessed in real environments with low overlap, specifically on the 3DMatch and 3DLoMatch datasets.

The 3DMatch dataset comprises a total of 62 scenes, distributed for training (46 scenes), validation (8 scenes), and testing (8 scenes) purposes. The 3DLoMatch dataset is a more challenging dataset derived from 3DMatch. In the original 3DMatch dataset, only point cloud pairs with an overlapping rate greater than 30% are employed for testing. In contrast, the testing set of 3DLomatch includes point cloud pairs with an overlapping rate ranging between 10% and 30%. Preprocessed training data are utilized, and its performance is evaluated using the established 3DMatch protocol.

In line with previous assessments, the performance of our algorithm is measured using three distinct metrics:

- Interior Point Ratio (IR): this metric quantifies the proportion of hypothetical correspondences with residuals falling below a predetermined threshold (e.g., 0.1 m) under the ground truth transformation;
- Feature Matching Recall (FMR): FMR denotes the fraction of point cloud pairs wherein the interior point ratio surpasses a specified threshold (e.g., 5%);

- Matching Recall (RR): RR involves evaluating the fraction of point cloud pairs exhibiting transformation errors below a given threshold (e.g., Root Mean Square Error < 0.2 m).

Experiments were performed identically on data from FCGF [33], D3feat, Predator, Cofinet and GeoTransformer (data obtained from the paper). As can be seen from Table 4, our model achieves the best performance in all three indicators. In 3DLoMatch (Table 5), compared to GeoTransformer, the FMR indicator is slightly insufficient. This is because when the point cloud overlap rate is too low, although multi-level sampling increases the number of point cloud pairs, the number of tasks with high confidence in the total registration tasks will decrease. Figure 3a,b represents the registration results of 3DMatch, and Figure 3c–e represents the registration results of 3DLoMatch. The algorithm achieves good registration results in both data sets.

Table 4. Registration results on 3DMarch.

Model	IR (%)	FMR (%)	RR (%)
FCGF [33]	48.7	97.0	83.3
D3feat [14]	40.4	94.5	83.4
Predator [12]	57.1	96.5	90.6
CoFiNet [15]	51.9	98.1	88.4
GeoTransformer [18]	70.3	97.7	91.5
ours	72.5	98.5	93.0

Table 5. Registration results on 3DLoMarch.

Model	IR (%)	FMR (%)	RR (%)
FCGF [33]	17.2	74.2	38.2
D3feat [14]	14.0	67.0	46.9
Predator [12]	28.3	76.3	61.2
CoFiNet [15]	26.7	83.3	64.2
GeoTransformer [18]	43.3	88.1	74.0
ours	45.6	87.2	75.3

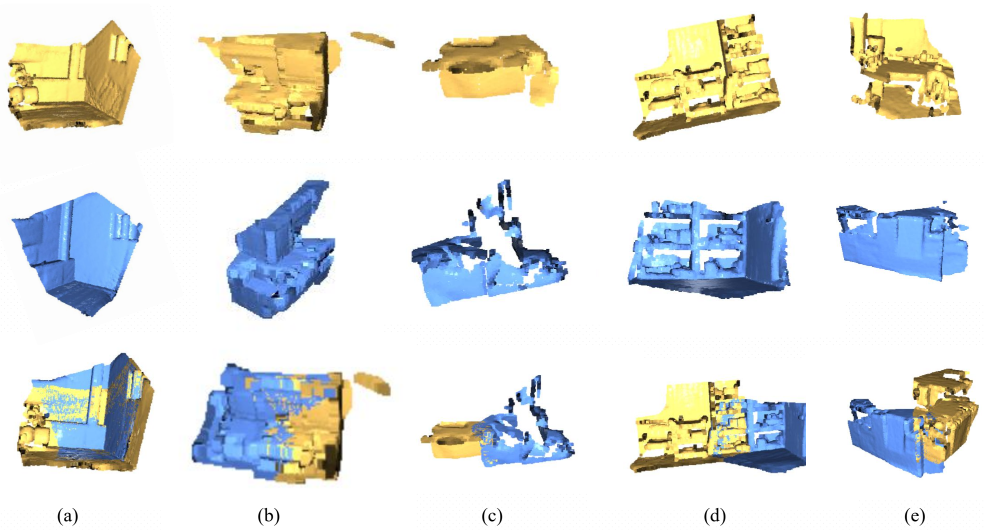


Figure 3. Results of 3DMatch and 3DLoMatch experiments. (**a**,**b**) are from 3DMatch, while (**c**–**e**) are from 3DLoMatch.

4.3. Outdoor Benchmark: KITTI Odometry

The KITTI odometry dataset encompasses 11 sequences capturing diverse outdoor driving scenarios, all of which are captured using LiDAR technology. Our utilization of this dataset is distributed as follows: sequences 0 to 5 are designated for training purposes, sequences 6 and 7 serve as validation sets, and sequences 8 to 10 constitute the testing data. In alignment with established practices in prior research, we adhere to the stipulation that only point cloud pairs separated by a minimum distance of 10 m are considered for evaluation.

Consistent with established practices in earlier studies, our performance evaluation hinges on three critical metrics:

- Relative Rotation Error (RRE): this metric quantifies the geodesic distance between the rotated matrix derived from our method and the corresponding ground truth rotated matrix;
- Relative Translation Error (RTE): RTE computes the Euclidean distance between the rotated matrices and the ground truth translation vectors;
- Recall to Alignment (RR): RR is a comprehensive metric reflecting the fraction of point-cloud pairs wherein both the RRE and RTE fall below specified thresholds (e.g., RRE < 5° and RTE < 2 m).

As shown in Table 6, our model is compared with [12,14,15,18,33]. In comparison to the real environment on the ground, we ensured similar displacement errors and rotation errors. Compared to other models, our metrics do not open a large gap, but still demonstrate the good generalizability of our model in outdoor environments. The registration effect is shown in the Figure 4.

Table 6. Registration results on KITTI odometry.

Model	RRE (°)	RTE (m)	RR (%)
FCGF [33]	0.30	9.5	96.6
D3Feat [14]	0.30	7.2	99.8
Predator [12]	0.27	6.8	99.8
CoFiNet [15]	0.41	8.2	99.8
GeoTransformer [18]	0.24	6.8	99.8
ours	0.218	5.4	99.8

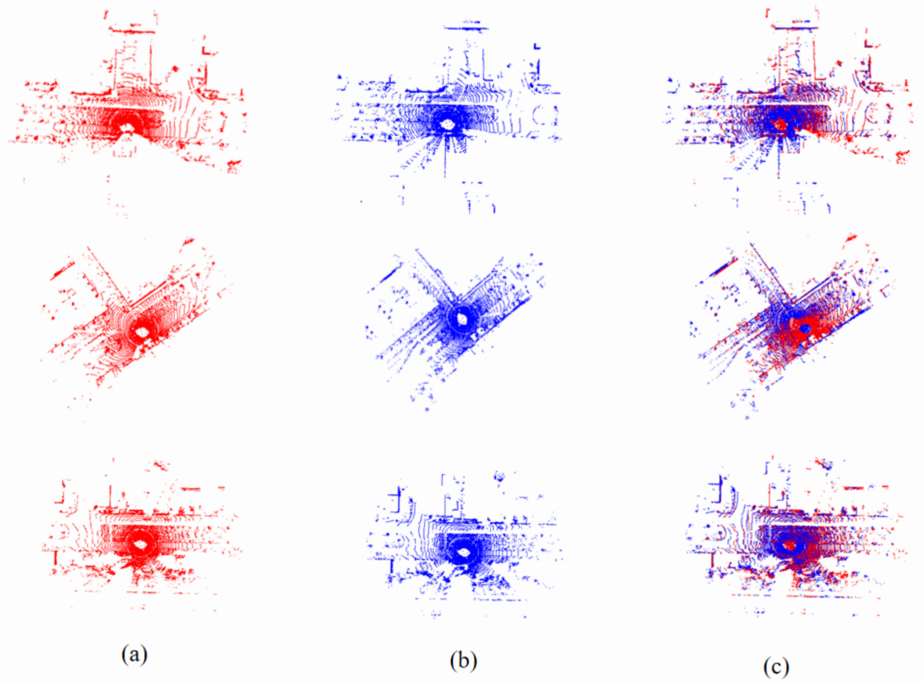


Figure 4. Results of KITTI experiments. (**a**,**b**) Point clouds collected using different radar positions and (**c**) is the result after registration.

4.4. Ablation Experiment

To illustrate the influence of each component on network performance, an ablation study is conducted in this section. Various modules within the network are systematically added and removed, allowing for an evaluation of their respective contributions to the final matching performance. Experiments are conducted on partially visible point cloud with noise. For easier comparison of results, we selected relative rotation error (RRE), relative translation error (RTE), and root mean square error (RMSE) as measurement standards. Experimental results (Table 7) show that a single graph convolution module can improve some accuracy, but there is still a gap with Transformer. After adding the cross-attention mechanism, our module indicators are already better than Transformer. The addition of Multi-channel further increased the rotation error and translation error by 7.3% and 5.8%. In addition, experimental results prove that our improved method achieves performance improvement compared to the baseline model and has good versatility.

Table 7. Ablation experiments.

Baseline	Trans-Former	Self-gnn	CA Blocks	Multi-Channel	RRE (°)	RTE (m)	RMSE
✓					2.154	0.033	0.026
✓	✓				1.577	0.018	0.017
✓		✓			1.723	0.029	0.021
✓		✓	✓		1.554	0.017	0.016
✓		✓	✓	✓	1.44	0.016	0.015

5. Conclusions

This paper proposes a novel network to solve the problem of point cloud registration in low-overlap environments. Compared with previous work, our model uses multi-layer patches to enrich correspondences and can still extract reliable correspondences from disordered point clouds in the environment of sparse keypoints. In addition, the template point matching module enhances the contextual features of patches through graph convolutional neural networks and multiple sub-attention mechanisms, guiding the model to match nodes with nearby regions and narrowing the search space for subsequent refinement. Experiments on multiple datasets show that our proposed method is very robust and still has high general-purpose capabilities on outdoor datasets.

Author Contributions: Conceptualization, J.Q.; methodology, J.Q.; software, J.Q.; validation, J.Q. and D.T.; formal analysis, D.T.; investigation, J.Q. and D.T.; resources, D.T.; data curation, J.Q. and D.T.; writing—original draft preparation, J.Q.; writing—review and editing, J.Q. and D.T.; visualization, J.Q.; supervision, D.T.; project administration, D.T.; funding acquisition, D.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Hunan Provincial Regional Joint Fund (2023JJ50130).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are not currently publicly available but are available from the authors upon reasonable request.

Conflicts of Interest: The authors declare no conflict of interest

References

1. Teng, S.; Hu, X.; Deng, P.; Li, B.; Li, Y.; Ai, Y.; Yang, D.; Li, L.; Xuanyuan, Z.; Zhu, F.; et al. Motion planning for autonomous driving: The state of the art and future perspectives. *IEEE Trans. Intell. Veh.* **2023**, *9*, 3692–3711.
2. Li, J.; Gao, W.; Wu, Y.; Liu, Y.; Shen, Y. High-quality indoor scene 3D reconstruction with RGB-D cameras: A brief review. *Comput. Vis. Media* **2022**, *8*, 369–393. [CrossRef]
3. Kazerouni, I.A.; Fitzgerald, L.; Dooly, G.; Toal, D. A survey of state-of-the-art on visual SLAM. *Expert Syst. Appl.* **2022**, *205*, 117734. [CrossRef]

4. Sun, X.; Wang, S.; Wang, M.; Cheng, S.S.; Liu, M. An advanced LiDAR point cloud sequence coding scheme for autonomous driving. In Proceedings of the 28th ACM International Conference on Multimedia, Seattle, WA, USA, 12–16 October 2020; pp. 2793–2801.
5. Sun, X.; Wang, M.; Du, J.; Sun, Y.; Cheng, S.S.; Xie, W. A Task-Driven Scene-Aware LiDAR Point Cloud Coding Framework for Autonomous Vehicles. *IEEE Trans. Ind. Inform.* **2022**, *19* , 8731–8742. [CrossRef]
6. Besl, P.J.; McKay, N.D. Method for registration of 3-D shapes. In *Proceedings of the Sensor fusion IV: Control Paradigms and Data Structures, Boston, MA, USA, 12–15 November 1991*; SPIE: Cergy-Pontoise, France; Volume 1611, pp. 586–606.
7. Zhou, Q.Y.; Park, J.; Koltun, V. Fast global registration. In *Proceedings of the Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016*; Proceedings, Part II 14; Springer: Berlin/Heidelberg, Germany , 2016; pp. 766–782.
8. Aoki, Y.; Goforth, H.; Srivatsan, R.A.; Lucey, S. Pointnetlk: Robust & efficient point cloud registration using pointnet. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 7163–7172.
9. Sarode, V.; Li, X.; Goforth, H.; Aoki, Y.; Srivatsan, R.A.; Lucey, S.; Choset, H. Pcrnet: Point cloud registration network using pointnet encoding. *arXiv* **2019**, arXiv:1908.07906.
10. Wang, Y.; Solomon, J.M. Deep closest point: Learning representations for point cloud registration. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Long Beach, CA, USA, 15–20 June 2019; pp. 3523–3532.
11. Wang, Y.; Solomon, J.M. Prnet: Self-supervised learning for partial-to-partial registration. *Adv. Neural Inf. Process. Syst.* **2019**, *32* .
12. Huang, S.; Gojcic, Z.; Usvyatsov, M.; Wieser, A.; Schindler, K. Predator: Registration of 3d point clouds with low overlap. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 4267–4276.
13. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3431–3440.
14. Bai, X.; Luo, Z.; Zhou, L.; Fu, H.; Quan, L.; Tai, C.L. D3feat: Joint learning of dense detection and description of 3d local features. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 6359–6367.
15. Yu, H.; Li, F.; Saleh, M.; Busam, B.; Ilic, S. Cofinet: Reliable coarse-to-fine correspondences for robust pointcloud registration. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 23872–23884.
16. Li, X.; Han, K.; Li, S.; Prisacariu, V. Dual-resolution correspondence networks. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 17346–17357.
17. Zhou, Q.; Sattler, T.; Leal-Taixe, L. Patch2pix: Epipolar-guided pixel-level correspondences. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 4669–4678.
18. Qin, Z.; Yu, H.; Wang, C.; Guo, Y.; Peng, Y.; Xu, K. Geometric transformer for fast and robust point cloud registration. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 11143–11152.
19. Yookwan, W.; Chinnasarn, K.; So-In, C.; Horkaew, P. Multimodal Fusion of Deeply Inferred Point Clouds for 3D Scene Reconstruction Using Cross-Entropy ICP. *IEEE Access* **2022**, *10*, 77123–77136. [CrossRef]
20. Vizzo, I.; Guadagnino, T.; Mersch, B.; Wiesmann, L.; Behley, J.; Stachniss, C. Kiss-icp: In defense of point-to-point icp–simple, accurate, and robust registration if done the right way. *IEEE Robot. Autom. Lett.* **2023**, *8*, 1029–1036. [CrossRef]
21. Liu, S.; Gao, D.; Wang, P.; Guo, X.; Xu, J.; Liu, D.X. A depth-based weighted point cloud registration for indoor scene. *Sensors* **2018**, *18*, 3608. [CrossRef] [PubMed]
22. Rusu, R.B.; Blodow, N.; Beetz, M. Fast point feature histograms (FPFH) for 3D registration. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; pp. 3212–3217.
23. Salti, S.; Tombari, F.; Di Stefano, L. SHOT: Unique signatures of histograms for surface and texture description. *Comput. Vis. Image Underst.* **2014**, *125*, 251–264. [CrossRef]
24. Wei, T.; Patel, Y.; Shekhovtsov, A.; Matas, J.; Barath, D. Generalized differentiable RANSAC. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 2–6 October 2023; pp. 17649–17660.
25. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660.
26. Phan, A.V.; Le Nguyen, M.; Nguyen, Y.L.H.; Bui, L.T. Dgcnn: A convolutional neural network over large-scale labeled graphs. *Neural Netw.* **2018**, *108*, 533–543. [CrossRef] [PubMed]
27. Yew, Z.J.; Lee, G.H. Rpm-net: Robust point matching using learned features. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11824–11833.
28. Zhao, H.; Wei, S.; Shi, D.; Tan, W.; Li, Z.; Ren, Y.; Wei, X.; Yang, Y.; Pu, S. Learning Symmetry-Aware Geometry Correspondences for 6D Object Pose Estimation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 2–6 October 2023; pp. 14045–14054.
29. Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; Xiao, J. 3d shapenets: A deep representation for volumetric shapes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1912–1920.

30. Zeng, A.; Song, S.; Nießner, M.; Fisher, M.; Xiao, J.; Funkhouser, T. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1802–1811.
31. Mei, G.; Huang, X.; Zhang, J.; Wu, Q. Overlap-guided coarse-to-fine correspondence prediction for point cloud registration. In Proceedings of the 2022 IEEE International Conference on Multimedia and Expo (ICME), Taipei, Taiwan, 18–22 July 2022; pp. 1–6.
32. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The kitti dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [CrossRef]
33. Choy, C.; Park, J.; Koltun, V. Fully convolutional geometric features. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 8958–8966.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

sensors

Article

Dimensioning Cuboid and Cylindrical Objects Using Only Noisy and Partially Observed Time-of-Flight Data

Bryan Rodriguez *, Prasanna Rangarajan, Xinxiang Zhang and Dinesh Rajan

Department of Electrical and Computer Engineering, Lyle School of Engineering, Southern Methodist University, Dallas, TX 75205, USA; prangara@mail.smu.edu (P.R.); xinxiang@mail.smu.edu (X.Z.); rajand@lyle.smu.edu (D.R.)
* Correspondence: brodrigu@mail.smu.edu

Abstract: One of the challenges of using Time-of-Flight (ToF) sensors for dimensioning objects is that the depth information suffers from issues such as low resolution, self-occlusions, noise, and multipath interference, which distort the shape and size of objects. In this work, we successfully apply a superquadric fitting framework for dimensioning cuboid and cylindrical objects from point cloud data generated using a ToF sensor. Our work demonstrates that an average error of less than 1 cm is possible for a box with the largest dimension of about 30 cm and a cylinder with the largest dimension of about 20 cm that are each placed 1.5 m from a ToF sensor. We also quantify the performance of dimensioning objects using various object orientations, ground plane surfaces, and model fitting methods. For cuboid objects, our results show that the proposed superquadric fitting framework is able to achieve absolute dimensioning errors between 4% and 9% using the bounding technique and between 8% and 15% using the mirroring technique across all tested surfaces. For cylindrical objects, our results show that the proposed superquadric fitting framework is able to achieve absolute dimensioning errors between 2.97% and 6.61% when the object is in a horizontal orientation and between 8.01% and 13.13% when the object is in a vertical orientation using the bounding technique across all tested surfaces.

Keywords: 3D scanning; 3D metrology; point cloud processing; Time-of-Flight sensors

1. Introduction

This work presents a method for dimensioning cuboid and cylindrical objects from noisy and partially occluded point cloud data that are acquired using a Time-of-Flight (ToF) sensor. Over the last decade, there has been an increase in the types of applications where ToF sensors are used, such as 3D scanning [1–5], drone positioning [6–8], robotics [9–11], and logistics [12–14]. In applications such as metrology and logistics, the ability to accurately determine the dimensions of an object is critical, for example, for part picking, packaging, and estimating shipping costs and storage needs. The quality of the depth information that is provided using modern three-dimensional (3D) sensors depends on the underlying technology. Existing works on dimensioning objects use low-noise, high-resolution 3D sensors such as structured light, stereo vision, and LiDAR to generate depth information [15–17]. These 3D sensors each have different tradeoffs and limitations that may not make them suitable for certain applications. For example, stereo vision systems typically have high software complexity, low depth accuracy, weak low-light performance, and limited range [18,19]. As another example, structured lights typically have high material costs, slow response times, and weak bright-light performance [18,19]. Compared to other 3D sensor technologies, ToF sensors provide a low-cost, compact design with low software complexity, fast response time, and good low-light and bright-light performance that can be used in real-time for generating depth information [18,19]. Despite these advantages, ToF sensing suffers from noise artifacts and issues such as multipath interference, which can distort the shape and size of objects in point clouds captured with a

ToF sensor. Figure 1 illustrates an example of a point cloud of the side profile of a regular box without and with multipath interference. As shown in Figure 1, multipath interference distorts the profile of the object such that the planar surfaces appear curved. The profile of a cylindrical object also experiences the same type of distortion. The distorted appearance of the object poses a challenge when trying to determine the dimensions of the object. In addition, ToF sensors also suffer from other issues such as low resolution, flying pixels, and self-occlusions, which further makes metrology challenging [20].

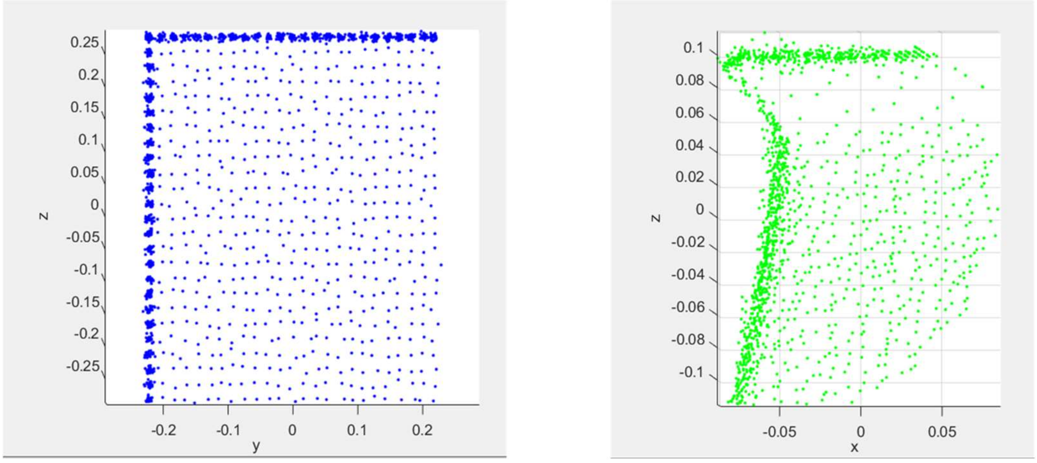


Figure 1. Point cloud of the side profile of a cuboid object without multipath interference (**left**) and with multipath interference (**right**).

A common approach to reducing the effect of multipath interference is to place the ToF sensor directly above an object that is placed flat on a ground surface in a top-view fronto-parallel configuration [21,22]. In this configuration, only the top surface of an object is visible to the ToF sensor. An example of a cuboid object in such a top-view fronto-parallel configuration is shown in Figure 2. In this configuration, the *x*- and *y*-dimensions of the object can be readily determined with respect to the *x*–*y* ground plane. The *z*-dimension can also be readily determined by taking the difference in depth measurements between the top surface of the object and the ground plane surface. In this configuration, since the interface between the object and the ground surface is not visible, the effect of multipath interference between the object and the ground surface is greatly reduced. However, this configuration is only feasible for a limited number of environments and applications.

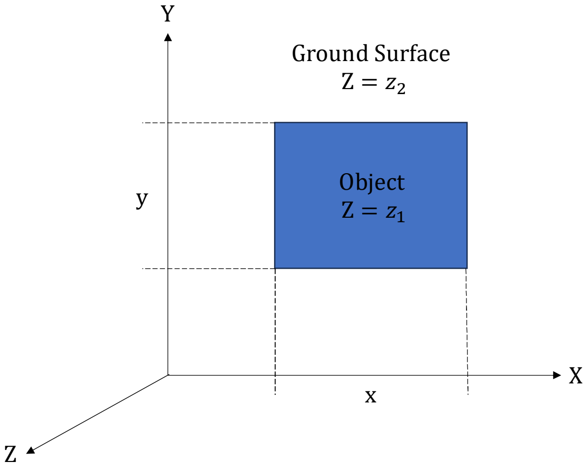


Figure 2. Top-view fronto-parallel configuration of a cuboid object on a ground plane surface.

Using a perspective view of an object proves the most flexibility for applications, but processing point cloud data captured using a perspective view poses challenges due to the presence of self-occlusions, noise, and multipath interference [20]. The amount of noise and multipath interference is scene-specific and depends on the ground surface material an object is resting on and the pose of the object. One approach for dimensioning cuboid objects involves using a plane fitting to identify the various surfaces of a box [23]. This approach typically requires that at least three surfaces of the box are visible to the ToF sensor and uses a RANSAC algorithm [24] on the point cloud of a box for detecting planes that correspond with surfaces of the box. Notably, this approach is limited to cuboid objects and cannot be applied to non-cuboid objects. Our results demonstrate that the number of points that are present on the surface of an object depends on the pose of the object with respect to the ToF sensor. This plane-fitting approach begins to breakdown as the number of points decreases on the surfaces of the object.

In this work, we developed a superquadric fitting framework for dimensioning cuboid and cylindrical objects using point cloud data that are generated with a ToF sensor that has a perspective view of an object. Our approach allows for the dimensioning of objects without requiring a top view of an object and without requiring that three sides of the object be visible to the ToF sensor. Previous works in robotic grasping applications have implemented a type of superquadric fitting to point cloud data for determining the general orientation and size of an object that is to be picked up with a robotic hand [25–28]. However, these works focused on obtaining rough dimensions for an object for grasping and did not attempt to quantify how accurately the dimensions of the object can be determined in various environments and orientations. Other existing works have employed superquadric fitting to point cloud data for identifying and classifying objects [29–31]. The focus of these works is to generally classify objects. These works also do not attempt to quantify how accurately the dimensions of an object can be determined. Further, these works typically rely on point cloud data that are obtained using other types of 3D sensor technologies, which do not suffer from the same types of noise artifacts and issues as a ToF sensor. These works do not suggest or provide any evidence that their approaches can be similarly applied to the same type of noisy point cloud data that are obtained from a ToF sensor. As discussed above, there is a tradeoff between the quality of the point cloud data that can be obtained and the low-cost, compact design of a ToF sensor that can be used in real-time for generating depth information.

Our proposed framework uses a non-linear least squares regression to determine a superquadric shape that best fits the point cloud data for an object while limiting the overgrowth of the superquadric shape. Our experiments show that during the fitting process, the dimensions of the superquadric shape tend to overgrow in the direction where data points are missing for the surfaces of an object due to self-occlusion. This overgrowth leads to significant errors in the dimensions of the object. A previous study by Quispe et al. also noted that superquadric models tend to overgrow during the superquadric fitting process [32]. When a superquadric model overgrows during fitting, the dimensions of the superquadric shape extend beyond the point cloud of the object. This type of overgrowth leads to inaccurate dimension estimates. Quispe et al. used an approach that was inspired by Bohg et al. that involves identifying a symmetry plane within the point cloud of an object and then using projection to artificially generate surfaces that are missing within the point cloud [16,32]. In their approach, the point cloud data that are used are generated using stereovision and RGBD cameras, which do not suffer from the same type of noise issues (e.g., multipath interference) as point cloud data from a ToF sensor. This approach is not suitable for the types of noisy point clouds that are generated using a ToF sensor because multipath interference distorts the point cloud of objects and makes identifying the surfaces of the object more difficult.

This work contributes to the state of the art by (1) developing a framework for dimensioning cuboid and cylindrical objects using enhanced superquadric fitting techniques and noisy point cloud data that are generated with a single ToF sensor. Our results show that a traditional superquadric fitting technique alone are insufficient for accurately determining the dimensions of an object using point cloud data that suffer from issues such as low resolution, self-occlusions, noise, and multipath interference. Our enhanced superquadric fitting techniques include bounding techniques for limiting superquadric overgrowth as well as considerations for the orientation of an object; (2) quantifying the accuracy for dimensioning cuboid and cylindrical objects on various types of ground surfaces using the noisy and partially observed point cloud data from a ToF sensor. The ground surfaces considered in this work include aluminum foil, black posterboard, white posterboard, and black felt. Each of these ground surfaces has different levels of infrared reflectivity; (3) quantifying the accuracy for dimensioning cuboid and cylindrical objects with different rotation angles and orientations with respect to the ToF sensor; and (4) quantifying the accuracy for dimensioning cuboid and cylindrical objects using various techniques for limiting overgrowth when fitting superquadric models. In applications such as logistics, the ability to accurately dimension objects is critical for operations like object grasping, packaging, storing, and transportation [33–37]. The tolerances for dimensioning errors vary from system to system. As these systems are further developed, their tolerances are typically reduced to optimize efficiency, object handling, and packaging [33–37]. As such, it becomes increasingly important to understand and quantify the performance and accuracy of object dimensioning techniques and the various factors that affect their performance. As discussed above, the presence of issues such as low resolution, self-occlusions, noise, and multipath interference all negatively impact and limit the usage of traditional techniques for dimensioning objects using point cloud data. This work contributes to the state of the art by quantifying the performance and accuracy of our proposed framework compared to traditional techniques for dimensioning objects using point cloud data. In addition, this work further contributes by quantifying how various environmental factors, such as ground surface material and object orientation, impact the performance and accuracy of our proposed framework.

This work uses a Texas Instrument TI OPT8241 ToF sensor for its experiments due to its widespread use in research and engineering applications. Since other ToF sensors operate using the same principles, which involve emitting and capturing reflected IR light, our framework for dimensioning cuboid and cylindrical objects using point cloud data from a ToF sensor can therefore also be generally applied to other types of ToF sensors since they all experience the same types of issues such as self-occlusions, noise, and multipath interference [20].

This paper is organized as follows: Section 2 discusses our proposed superquadric fitting framework for dimensioning cuboid and cylindrical objects. Section 3 discusses the experimental setup and the numerical results. Finally, Section 4 provides concluding remarks.

2. Methodology

As discussed above, point cloud data that are obtained from a single ToF sensor typically suffers from issues such as low resolution, self-occlusions, noise, and multipath interference. These issues tend to distort the shape and size of objects, which creates challenges for dimensioning objects using a ToF sensor. In this work, we propose an approach that overcomes these challenges by fitting a parametric model to the point cloud data. Given a set of point cloud data points (x_w, y_w, z_w) from a ToF sensor, our proposed approach uses non-linear least squares fitting to determine the parametric model that best fits the point cloud data. Our experiments show that directly applying a parametric fit to the point cloud data without any preprocessing results in large estimation errors. To address this issue, in our proposed framework, we preprocess the point cloud data using the following steps: ground plane rectification, ground plane segmentation, and reorienting the point cloud within a new local coordinate system before performing the initial pose estimation. Using this approach, the subsequent parametric fitting shows significantly lower dimensioning errors. Figure 3 provides an overview of our methodology.

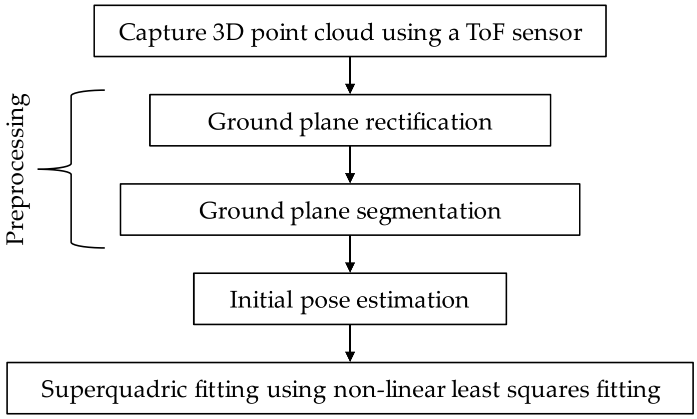

Capture 3D point cloud using a ToF sensor
Preprocessing
Ground plane rectification
Ground plane segmentation
Initial pose estimation
Superquadric fitting using non-linear least squares fitting

Figure 3. Process for performing parametric fitting for dimensioning of an object.

The key steps to the parametric fitting for dimensioning methodology are as follows: A ToF sensor is configured to capture a point cloud from a perspective view of an object within a scene. A ground plane rectification process is first performed to compensate for the perspective view of the ToF sensor. Ground plane segmentation is then performed to segment the object from the rest of the scene. An initial pose is then determined for the object, and the point cloud for the object is reoriented such that the object is axis-aligned and centered about a user-defined local origin. The reoriented point cloud is then fed into a superquadric fitting algorithm. As part of the fitting process, we use either a bounding technique or a mirroring technique to limit any overgrowth of the superquadric model. The bounding technique limits superquadric shape overgrowth by applying adaptive upper and lower bounds to the dimensions of the superquadric shape during the fitting process. The mirroring technique limits superquadric shape overgrowth by synthetically generating data points for the surfaces of the object that are missing due to self-occlusion. The mirroring technique generates a more complete point cloud representation of an object that reduces the number of missing surfaces, which would allow the superquadric shape to overgrow during the fitting. The object dimensions can then be obtained based on the determined parameters of the superquadric shape that is fitted to the point cloud.

2.1. Ground Plane Rectification

Figure 4 is an example of an intensity image of a scene with fiducial markers and a box positioned on top of a black felt surface. The dimensions of the box are labeled as a_1, a_2, and a_3. In this example, a_1 = 149 mm, a_2 = 223 mm, and a_3 = 286 mm. In our experiments, ArUco fiducial markers are initially used to determine the orientation of the ground plane with respect to the ToF sensor. An ArUco marker is a 2D binary-encoded fiduciary marker that can be used for camera pose estimation [38]. ArUco markers were selected due to their widespread use in various computer vision-based applications, such as robotics and automation. This approach does not require a pre-calibrated camera mounting system with respect to the object plane and is more robust and applicable to dynamic settings where a camera is mounted to a movable arm or robot system. Although ArUco markers were used in this work, a similar approach can be implemented using other suitable types of fiducial markers.

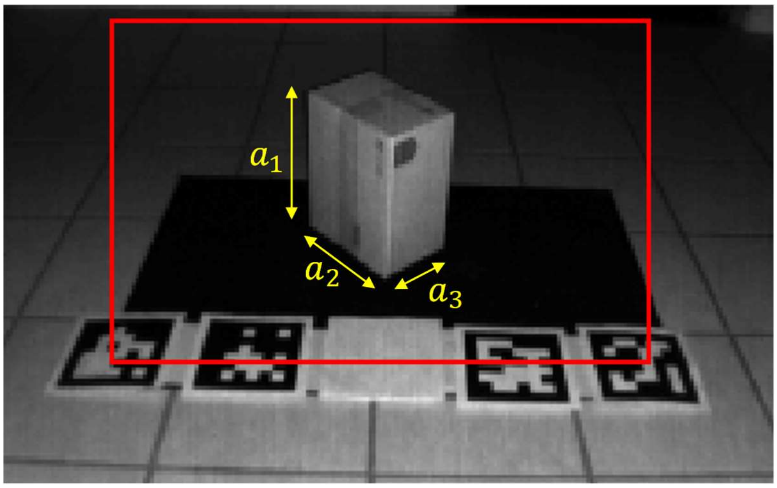


Figure 4. Intensity image of a scene with ArUco markers and a box positioned on a black felt surface. The region of interest (ROI) for our object is represented by the red bounding box.

To determine the ground plane orientation, an ArUco marker is placed on the ground plane within the field of view of the ToF sensor. We then capture and process an intensity image of the scene using the OpenCV libraries [39] to detect the presence and orientation of the ArUco marker. The orientation of the ArUco marker provides information about the orientation of the ground plane with respect to the ToF sensor. Once the orientation of the ground plane is determined, the point cloud for the entire scene is then rotated such that the ground plane is aligned with a horizontal *x*–*y* plane in our coordinate system. Although only one ArUco marker is required to determine the orientation of the ground plane, we use four markers for redundancy. The ArUco markers are also used to identify and crop the region of interest by positioning the object between the outermost ArUco markers. Figure 5 is an example of a point cloud after ground plane rotation correction. As shown in Figure 5, the ground plane in our point cloud data are substantially parallel with the horizontal *x*–*y* plane after ground plane rotation.

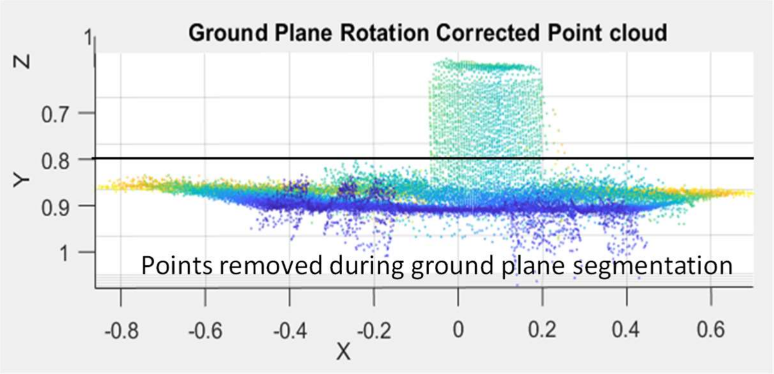


Figure 5. Front view of a point cloud of a scene with a box on a ground plane surface after ground plane correction in meter units. An offset threshold for the ground plane segmentation process is represented by the solid black horizontal line. The color of a data point in the point cloud corresponds with a distance between the ToF sensor and a surface in the scene. Darker colors (e.g., dark blue) represent surfaces closer to the ToF sensor and lighter colors (e.g., yellow) represent surfaces further away from the ToF sensor.

2.2. Ground Plane Segmentation

Following ground plane rectification, the position of the ground plane is known. Thresholding is then performed using an offset threshold value to segment the object of interest from the ground plane. As shown in Figure 5, the point cloud for the ground plane appears noisy primarily due to multipath interference near the interface between the ground plane and the faces of the object. In this example, additional noise is also caused by the fiducial markers. The noise from the fiducial markers and the multipath interference between the ground plane and the object are removed during segmentation, and the remaining point cloud corresponds to the object of interest. Any residual multipath interference can be reduced by increasing the offset threshold value. The tradeoff for this approach is that increasing the offset threshold value reduces the number of points on the object that are available for the parametric fitting process. Since the offset threshold value is known, this value is added back later as a correction term to one of the dimensions of the object after performing the parametric fitting. Figure 6 shows an example of the remaining point cloud data for a box after performing ground plane segmentation.

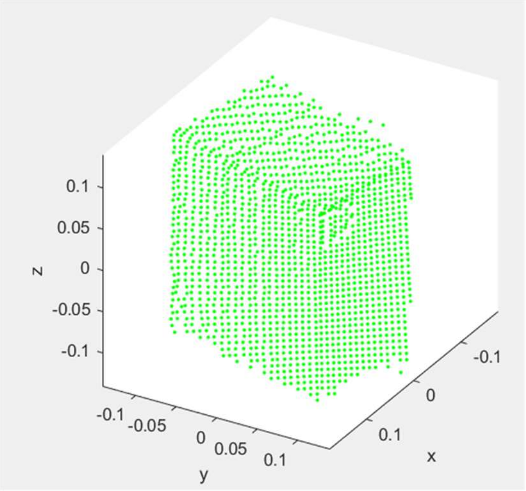


Figure 6. Example of the remaining point cloud data for a box after performing ground plane segmentation in meter units.

2.3. Initial Pose Estimation

To determine the initial pose of the object, the remaining point cloud of the object is reoriented such that the object is centered about a user-defined origin and axis aligned. In this work, this reorientation is performed by flattening the point cloud into the direction of the ground plane to form a top-view representation of the object. Figure 7 illustrates an example of the result of the flattening process for the point cloud with respect to the ground plane. By flattening the point cloud in this manner, dense clusters of point clouds will appear, which correspond with the edges of the object. Once the point cloud has been flattened, a RANSAC ("RANdom Sample Consensus") algorithm [24] is used to identify an edge of the object by fitting a line to one of the edges of the point cloud. The RANSAC process first identifies the dense clusters of points that correspond with the edges of the object and then fits a line to one of these clusters of data points. In the example shown in Figure 7, the line that was determined from the RANSAC process is represented as a solid blue line. Once the orientation of an edge of the object is known, the point cloud is rotated such that the object edges are axis-aligned. First, an angle is determined between the line that was determined from the RANSAC process and either the x- or y-axis of the coordinate system. Then, the entire point cloud is rotated about the vertical z-axis with the determined angle to axis align the point cloud with the x–y plane. The axis-aligned point cloud is then shifted such that the center of the point cloud is at the local origin. An example of the result of this process is also shown in Figure 7.

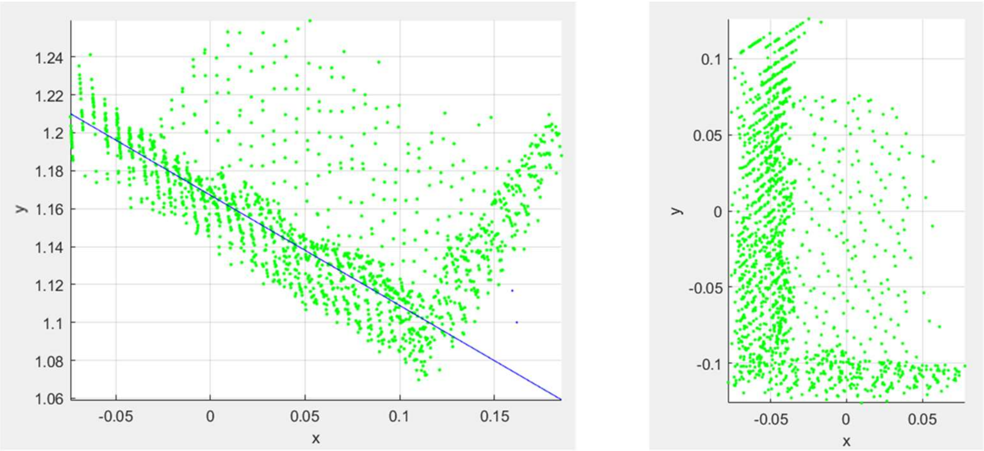


Figure 7. Example of reorienting the point cloud data for a box before (**left**) and after axis alignment (**right**) in meter units. In the left plot, the blue line corresponds with the orientation for an edge of the box that was determined from the RANSAC process.

Once the point cloud is centered at the origin, the initial rotation parameters (ϕ, θ, ψ) and translation parameters (p_x, p_y, p_z) are set to zero. Although the fitting process is capable of solving for non-zero translation and rotation parameters, our experiments showed improvements in terms of accuracy and speed by reorienting our point cloud and initially setting these parameters to zero. The fitting process will later adjust the translation and rotation parameters to best fit the point cloud data. The initial dimension parameters (a_1, a_2, a_3) of the object are determined based on the difference between the minimum and maximum values of the point cloud along each axis.

2.4. Limiting Superquadric Growth

At this point, we can fit a superquadric model to the remaining point cloud data. However, our experiments showed poor performance resulting from overgrowth in the direction where data points are missing for the surfaces of an object due to self-occlusion. In the first set of experiments, the fitting was performed on the axis-aligned point cloud. In these experiments, adaptive upper and lower bounds were used to limit overgrowth on the dimensions of the superquadric model that was generated. In our experiments, tolerances of 5%, 2%, and 1% are applied to the dimensions estimated during the initial pose estimation process to determine the bounds. The upper and lower limits were used because the point cloud of the object is incomplete due to self-occlusions. For example, with cuboid objects, the point cloud data only have points on the surfaces of the box that are visible to the ToF sensor and do not include points that represent the bottom or the backside of the box. In some instances, when a superquadric model is fit to the point cloud with partial data, the superquadric grows beyond the points in the point cloud where the surface of an object is not represented. This occurs because the minima in Equation (2) does not guarantee returning the smallest superquadric model that fits the point cloud data. In a second set of experiments, the fitting process is performed on a mirrored version of the axis-aligned point cloud. The mirrored point cloud is generated by creating a duplicate of the axis-aligned point cloud, flipping it 180° vertically about its centroid and the x-axis, and then rotating it 90° about the vertical z-axis. The duplicate point cloud is then merged with the original axis-aligned point cloud to form the mirrored version of the point cloud. By using the mirrored point cloud, data points for any non-visible sides of the object are synthetically created, and bounds are no longer necessary to limit any overgrowth in the fitting process. An example of the mirroring technique is shown in Figure 8.

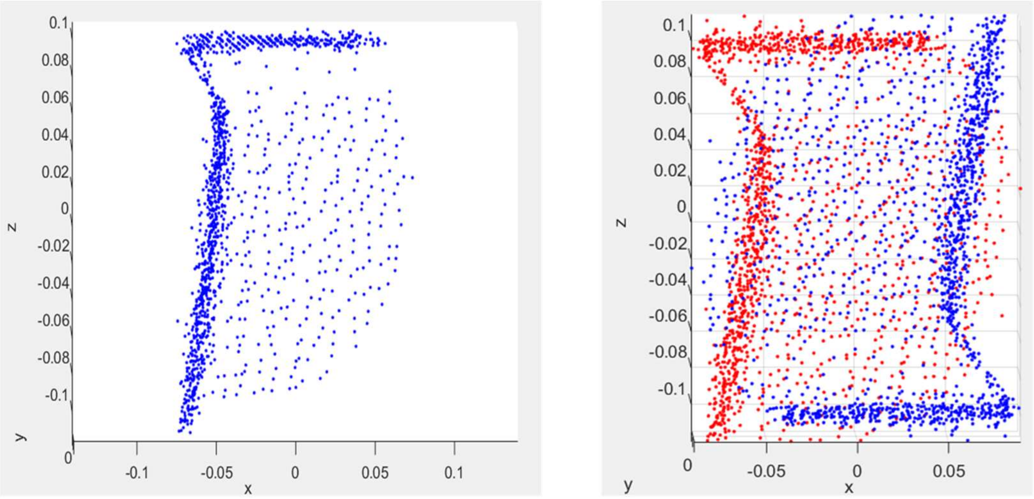


Figure 8. Profile view of a point cloud of a cuboid before (**left**) and after applying the mirroring technique for limiting superquadric overgrowth (**right**) in meter units. In the right plot, the red data points correspond with the initial point cloud of the cuboid and the blue data points correspond with the additional data points that were generated by applying the mirroring technique.

2.5. Non-Linear Least Squares Fitting

Our work involves performing a non-linear least squares fit of a superquadric shape to the point cloud of an object to determine the dimensions of the object. A superquadric is a parametric shape that has parameters that describe the size, shape, and pose of the superquadric [40,41]. In this work, a superquadric shape was selected because it can be morphed into a wide range of shapes [40]. By adjusting the shape parameters, the superquadric shape can be morphed into a range of symmetric objects, which include cuboids and cylinders. The implicit form of the superquadric equation that is used in this work is given using the inside–outside function F, which is defined as the following:

$$\begin{aligned} F(x_w, y_w, z_w) = & \left[\left(\frac{n_x x_w + n_y y_w + n_z z_w - p_x n_x - p_y n_y - p_z n_z}{a_1} \right)^{\frac{2}{\in_2}} \right. \\ & \left. + \left(\frac{o_x x_w + o_y y_w + o_z z_w - p_x o_x - p_y o_y - p_z o_z}{a_2} \right)^{\frac{2}{\in_2}} \right]^{\frac{\in_2}{\in_1}} \\ & + \left(\frac{a_x x_w + a_y y_w + a_z z_w - p_x a_x - p_y a_y - p_z a_z}{a_3} \right)^{\frac{2}{\in_1}} \end{aligned} \tag{1}$$

where variables (x_w, y_w, z_w) are the data points from the captured point cloud, (a_1, a_2, a_3) are the scaling dimensions along the x-, y-, and z-axis of the superquadric, respectively, $(\in_1, \in_2)$ are shape parameters, and $(n_x, n_y, n_z, o_x, o_y, o_z, a_x, a_y, a_z, p_x, p_y, p_z)$ are the twelve parameters of a homogenous transformation matrix that is the result of a rotation and a translation of the world coordinate plane [42,43]. The eleven parameters that define the position and orientation of a superquadric are defined as $\wedge$ = $\{a_1$, a_2, a_3, $\in_1$, $\in_2$, ϕ, θ, Ψ, p_x, p_y, $p_z\}$ [42,43].

Following our initial pose estimates, the initial rotation parameters (ϕ, θ, Ψ) and translation parameters (p_x, p_y, p_z) are set to zero, the initial object dimensions (a_1, a_2, a_3) are determined as the difference between the minimum and maximum values of the point cloud along each axis, and the shape parameters $(\in_1, \in_2)$ are initially set to an intermediate value of one. The final values of $\wedge$ are determined using a least squares minimization process. We perform a least squares minimization using the Levenberg–Marquardt algorithm [44] to recover the parameter set $\wedge$ that best fits the kth point, (x_k, y_k, z_k), in the point cloud. The following expression describes the minimization process:

$$\min_k \sum_{k=0}^{n} \left(\sqrt{a_1 a_2 a_3} \left(F^{\in_1}(x_k, y_k, z_k; \wedge) - 1 \right) \right)^2 \tag{2}$$

where the coefficient $\sqrt{a_1 a_2 a_3}$ is used to recover the smallest superquadric, and the exponent $\in_1$ promotes faster convergence by making the error metric independent of the shape parameters [43].

After determining the parameters (a_1, a_2, a_3) corresponding with the dimensions of the object, the offset threshold value that was used during the ground segmentation step is then added to the vertical dimension of the object to compensate for the portion of the point cloud that was removed during the ground plane segmentation process. Through this process, the full dimensions of the object are recovered.

Figure 9 is an example of a superquadric shape that is generated based on the parameters determined with the non-linear least squares fitting, which is overlaid with a corresponding object of interest (i.e., a box) within an intensity image.

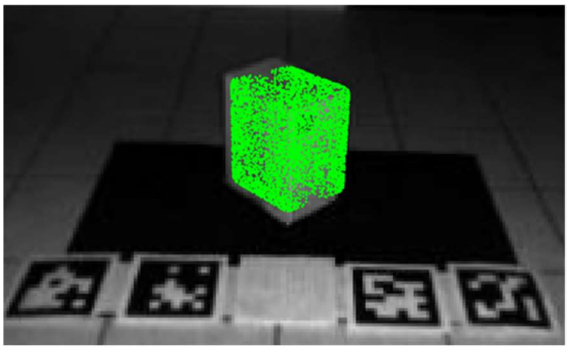


Figure 9. Example of a determined superquadric shape (shown in green) overlaid with a box in an intensity image. The superquadric shape is represented as a series of points corresponding with data points on the surface of the superquadric shape.

3. Experiments

3.1. Hardware Configuration

In our experiments, we use a single ToF sensor, TI OPT8241 [45], to generate depth information for a single object. This sensor is able to output both grayscale intensity images and point clouds. This sensor offers a resolution of 320 × 240 with a horizontal field-of-view of 74.4°. In each experiment, the object is located 1.5 m from the ToF sensor. The ToF sensor is positioned on a tripod with a downward perspective view of between 35 and 45° of the object. The physical dimensions of the boxes are between 122 mm and 365 mm in length. The cylinder has a height of 204 mm and a diameter of 155 mm. In each experiment, an object is placed on different ground plane surfaces that each have different levels of infrared reflectivity and multipath interference. The ground plane surfaces used in our experiment are aluminum foil, black poster board, white poster board, and black felt. For each ground plane surface, the object is rotated between angles of 30 and 75° with respect to the ToF sensor.

For capturing intensity images and point cloud data, we used a Voxel Viewer from Texas Instruments [46]. During the data collection period, an average of 400 frames of intensity images and point cloud data were collected for each experiment configuration. For implementing our framework, we used Matlab R2020b and OpenCV libraries [39,47]. The OpenCV libraries were primarily used for identifying fiducial markers in our ground plane rectification process.

3.2. Dimensioning Performance for Cuboid Objects Based on the Ground Plane Surface

Table 1 shows the average of absolute errors for each of the ground surfaces using various dimensioning techniques. As the box is rotated with respect to the ToF sensor, an error for each dimension of the box is computed. The absolute errors for each dimension are then averaged to determine the average of the absolute errors at each rotation angle of the box. The average of absolute errors is the average of the errors across all the rotation angles for the box from 30° to 75°.

Table 1. Dimension errors for a box using various surfaces and fitting techniques.

Method	Aluminum Foil	Black Posterboard	White Posterboard	Black Felt
Ellipsoid fit	99%	115%	79%	102%
Superquadric fit—no bounds or axis alignment	37%	228%	40%	33%
Superquadric fit—1% bounds with axis alignment	13%	10%	11%	13%
Superquadric fit—Mirroring	11%	15%	17%	13%

Table 1 shows that a traditional approach of fitting an ellipsoid to the point cloud results in large errors compared to the superquadric fit. Errors can be further reduced by using bounding or mirroring techniques to limit the overgrowth of the super-quadric model during the fitting process. Both techniques rely on fitting the superquadric shape after the point cloud is axis-aligned, which reduces the variation in the rotation parameters in the superquadric shape and improves performance. Our results show that the impact of multipath interference from the ground planes having different levels of infrared reflectivity is negligible using either technique.

3.3. Dimensioning Performance for a Cuboid Object Based on Object Orientation

Figure 10 shows the average of absolute errors for each ground plane surface at each rotation angle of the box using the bounding technique and the mirroring technique. Figure 11 shows the average of absolute errors for all of the ground plane surfaces at each rotation angle of the box with respect to the ToF sensor using the bounding technique and the mirroring technique.

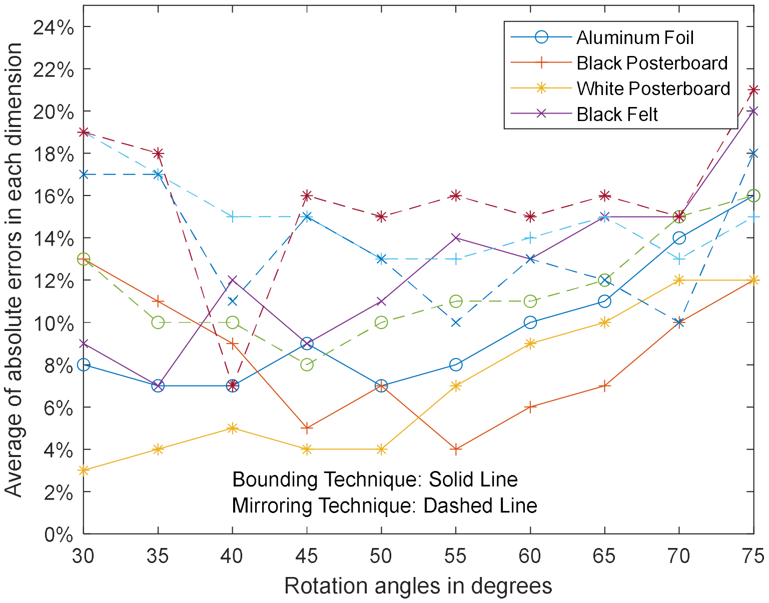


Figure 10. Dimension errors at each rotation angle of the box. Each marker type (i.e., o, +, *, x) corresponds with a ground surface material. Solid or dashed lines are used with the corresponding marker type based on whether the bounding technique or the mirroring technique was applied, respectively.

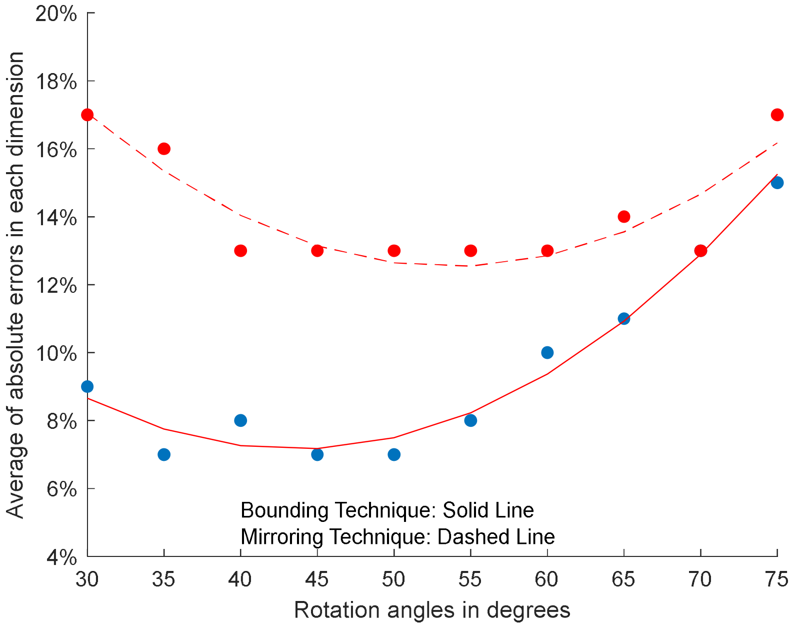


Figure 11. Average dimension errors of the box across all ground plane surfaces. The blue data points correspond with error measurements obtained using the bounding technique. The red data points correspond with error measurements obtained using the mirroring technique.

In Figures 10 and 11, the box is initially positioned at an angle of 45° with respect to the ToF sensor such that two side surfaces and the top surface of the object are visible to the ToF sensor. The box is then rotated about the vertical *z*-axis with respect to the ToF sensor to determine the effect of the rotation angle of the box with respect to the ToF sensor. The results from Figures 10 and 11 show that the smallest of the absolute errors for the various ground plane surfaces generally occur when the box is rotated about 45° with respect to the ToF sensor. In this orientation, both vertical faces are most visible to the ToF sensor, which results in more data points on the box surfaces being available for processing. As the box rotates away from 45°, one of the vertical sides of the box becomes less visible, resulting in fewer data points on the surfaces of the box being available for processing. Our results show that the average error increases as the number of data points decreases. In an extreme case, when the box is rotated head-on with the ToF sensor at 0° or 90°, only one vertical surface and the top surface are visible to the ToF sensor, which results in the fewest number of data points on the surfaces of the box. Although the average error increases as the number of data points on the surfaces of the box decreases, our experiments show that superquadric shapes can be used even when only two surfaces of the object are visible to the ToF sensor. While our framework can be applied when only one vertical surface and the top surface are visible to the ToF sensor, our results show that the average error further increases as the number of data points on the surfaces of the box decreases. In particular, the mirroring technique experiences a higher average error compared to the bounding technique since the mirroring technique relies on surface data points for synthetically creating non-visible sides of an object.

3.4. Dimensioning Performance for a Cuboid Object Using Bounding Technique

Table 2 shows the average of absolute errors for each of the ground plane surfaces with a box rotation of 45° with respect to the ToF sensor using the bounding technique and the mirroring technique for fitting a superquadric shape to the point cloud data. Our results show that the bounding technique can provide lower dimensioning errors compared to the mirroring technique. In our experiments, we observed that the mirroring technique tends to result in underfitting the superquadric shape to the point cloud data, which results in larger dimensioning errors.

Table 2. Dimension errors for various fitting techniques of a box rotation angle of 45°.

Method	Aluminum Foil	Black Posterboard	White Posterboard	Black Felt
Superquadric fitting—1% Bounds w/axis alignment	9%	5%	4%	9%
Superquadric fitting—Mirroring	8%	15%	16%	15%

3.5. Dimensioning Performance for a Cylindrical Object Using Bounding Technique

Based on the findings from the cuboid object experiments, we conducted similar experiments on a cylindrical object using the bounding technique since it provided better performance than the mirroring technique. Table 3 shows the average of absolute errors for each of the ground surfaces for different orientations of a cylinder using the bounding technique. As the cylinder is rotated with respect to the ToF sensor, an error for each dimension of the cylinder is computed. The absolute errors for each dimension are then averaged to determine the average of the absolute errors at each rotation angle of the cylinder. Table 3 shows that dimensioning the cylindrical object in the vertical orientation resulted in larger dimensioning errors compared to when the cylindrical object was in the horizontal orientation. In both orientations, our experiments showed an increase in the amount of missing data for cylindrical objects compared to cuboid objects due to the curved surfaces of the cylindrical object. These curved surfaces deflected more infrared light away from the ToF sensor, which resulted in less surface data being collected by the ToF sensor. Our experiments also showed that when the cylindrical object is in the horizontal orientation, the proximity of the ground plane to the curved surface of the cylinder reduces the amount of surface data that is lost compared to when the cylindrical object is in the vertical orientation. Figure 12 illustrates an example of the point cloud data for a cylindrical object and the corresponding superquadric fit to the point cloud data.

Table 3. Dimension errors for a cylinder using the bounding technique.

Cylinder Orientation	Aluminum Foil	Black Posterboard	White Posterboard	Black Felt
Horizontal	6.61%	6.35%	2.97%	6.29%
Vertical	8.01%	12.75%	12.87%	13.13%

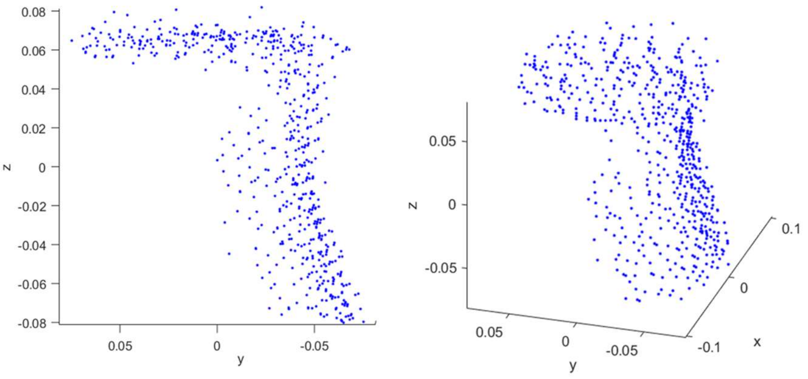


Figure 12. *Cont.*

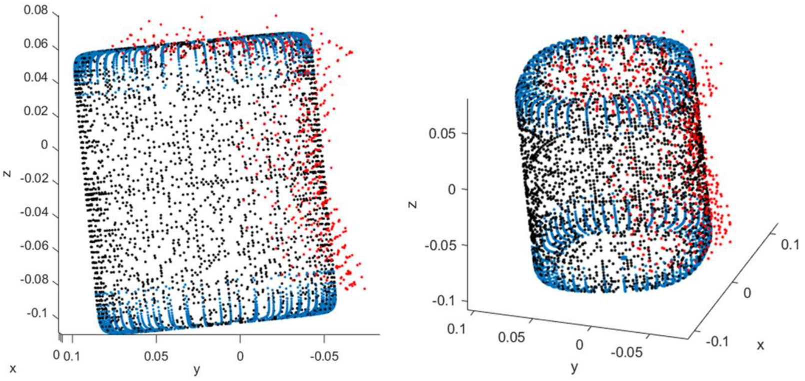


Figure 12. Profile view (**top left**) and perspective view (**top right**) of a point cloud for a cylindrical object and profile view of superquadric fit (**bottom left**) and perspective view of superquadric fit (**bottom right**) to the point cloud for the cylindrical object in meter units. In the bottom plots, the red data points correspond with the point cloud for the cylindrical object and the black and blue data points corresponds with data points on the surface of the superquadric shape that was determined from the superquadric fitting process.

4. Conclusions

In this work, we developed a framework that can be used for dimensioning cuboid and cylindrical objects from point cloud data generated using a ToF sensor despite the presence of noise artifacts and issues such as low resolution, self-occlusions, noise, and multipath interference. This work also quantifies the impact on the accuracy of dimensioning objects based on various model fitting techniques, the pose of an object, the shape of an object, and the ground surface material under an object.

Our results show that the performance of dimensioning a cuboid object increases when more surfaces and surface areas of the object are visible to the ToF sensor. Conversely, the performance of dimensioning a cuboid object decreases when fewer surfaces and surface areas are visible to the ToF sensor. In addition, the performance of dimensioning a cylinder object increases when the object is in a horizontal configuration as opposed to a vertical configuration. Our results also showed that dimensioning performance improves when a bounding technique is employed in conjunction with the parametric fitting process to reduce overgrowth of the superquadric shape. Notably, the bound-based approach provides better performance compared to a mirroring-based approach that synthetically creates missing point cloud information for an object.

This work can be extended to examine the use of multiple ToF sensors to further improve dimensioning accuracy. Future works may also extend the ability of parametric fitting to dimension more complex shapes with non-convex surfaces.

Author Contributions: Conceptualization, B.R., P.R. and D.R.; Data curation, B.R. and X.Z.; Formal analysis, B.R., P.R. and D.R.; Investigation, B.R., P.R., X.Z. and D.R.; Methodology, B.R., P.R. and D.R.; Project administration, P.R. and D.R.; Resources, P.R. and D.R.; Software, B.R.; Supervision, P.R. and D.R.; Validation, B.R. and D.R.; Visualization, B.R.; Writing—original draft, B.R.; Writing—review and editing, B.R. and D.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: We thank Texas Instruments for providing their support for this project. We also thank Duncan MacFarlane (Southern Methodist University) for providing lab space and Ashwini Subramanian (Southern Methodist University) for her initial technical assistance with this project.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Page, D.L.; Fougerolle, Y.; Koschan, A.F.; Gribok, A.; Abidi, M.A.; Gorsich, D.J.; Gerhart, G.R. SAFER vehicle inspection: A multimodal robotic sensing platform. In *Unmanned Ground Vehicle Technology VI, Proceedings of the Defense and Security, Orlando, FL, USA, 12–16 April 2004*; SPIE: St Bellingham, WA, USA, 2004; Volume 5422, pp. 549–561.
2. Chen, C.; Yang, B.; Song, S.; Tian, M.; Li, J.; Dai, W.; Fang, L. Calibrate Multiple Consumer RGB-D Cameras for Low-Cost and Efficient 3D Indoor Mapping. *Remote Sens.* **2018**, *10*, 328. [CrossRef]
3. Rodriguez, B.; Zhang, X.; Rajan, D. Synthetically Generating Motion Blur in a Depth Map from Time-of-Flight Sensors. In Proceedings of the 2021 17th International Conference on Machine Vision and Applications (MVA), Aichi, Japan, 25–27 July 2021; pp. 1–5.
4. Rodriguez, B.; Zhang, X.; Rajan, D. Probabilistic Modeling of Motion Blur for Time-of-Flight Sensors. *Sensors* **2022**, *22*, 1182. [CrossRef] [PubMed]
5. Rodriguez, B.; Zhang, X.; Rajan, D. Probabilistic Modeling of Multicamera Interference for Time-of-Flight Sensors. *Sensors* **2023**, *23*, 8047. [CrossRef] [PubMed]
6. Paredes, J.A.; Álvarez, F.J.; Aguilera, T.; Villadangos, J.M. 3D indoor positioning of UAVs with spread spectrum ultrasound and time-of-flight cameras. *Sensors* **2018**, *18*, 89. [CrossRef] [PubMed]
7. Mentasti, S.; Pedersini, F. Controlling the Flight of a Drone and Its Camera for 3D Reconstruction of Large Objects. *Sensors* **2019**, *19*, 2333. [CrossRef] [PubMed]
8. Jin, Y.-H.; Ko, K.-W.; Lee, W.-H. An Indoor Location-Based Positioning System Using Stereo Vision with the Drone Camera. *Mob. Inf. Syst.* **2018**, *2018*, 5160543. [CrossRef]
9. Pascoal, R.; Santos, V.; Premebida, C.; Nunes, U. Simultaneous Segmentation and Superquadrics Fitting in Laser-Range Data. *IEEE Trans. Veh. Technol.* **2014**, *64*, 441–452. [CrossRef]
10. Shen, S.; Mulgaonkar, Y.; Michael, N.; Kumar, V. Multi-Sensor Fusion for Robust Autonomous Flight in Indoor and Outdoor Environments with a Rotorcraft MAV. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 4974–4981.
11. Chiodini, S.; Giubilato, R.; Pertile, M.; Debei, S. Retrieving Scale on Monocular Visual Odometry Using Low-Resolution Range Sensors. *IEEE Trans. Instrum. Meas.* **2020**, *69*, 5875–5889. [CrossRef]
12. Correll, N.; Bekris, K.E.; Berenson, D.; Brock, O.; Causo, A.; Hauser, K.; Okada, K.; Rodriguez, A.; Romano, J.M.; Wurman, P.R. Analysis and Observations from the First Amazon Picking Challenge. *IEEE Trans. Autom. Sci. Eng.* **2016**, *15*, 172–188. [CrossRef]
13. Corbato, C.H.; Bharatheesha, M.; Van Egmond, J.; Ju, J.; Wisse, M. Integrating Different Levels of Automation: Lessons from Winning the Amazon Robotics Challenge 2016. *IEEE Trans. Ind. Inform.* **2018**, *14*, 4916–4926. [CrossRef]
14. Pardi, T.; Poggiani, M.; Luberto, E.; Raugi, A.; Garabini, M.; Persichini, R.; Catalano, M.G.; Grioli, G.; Bonilla, M.; Bicchi, A. A Soft Robotics Approach to Autonomous Warehouse Picking. In *Advances on Robotic Item Picking*; Springer: Cham, Switzerland, 2020; pp. 23–35.
15. Park, H.; Van Messem, A.; De Neve, W. Item Measurement for Logistics-Oriented Belt Conveyor Systems Using a Scenario-Driven Approach and Automata-Based Control Design. In Proceedings of the 2020 IEEE 7th International Conference on Industrial Engineering and Applications (ICIEA), Bangkok, Thailand, 16–18 April 2020; pp. 271–280. [CrossRef]
16. Bohg, J.; Johnson-Roberson, M.; León, B.; Felip, J.; Gratal, X.; Bergström, N.; Kragic, D.; Morales, A. Mind the gap—Robotic grasping under incomplete observation. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 686–693. [CrossRef]
17. Song, K.-T.; Ou, S.-Q. A Client-Server Architecture for Object Volume Measurement on a Conveyor Belt. In Proceedings of the 2019 12th Asian Control Conference (ASCC), Kitakyushu-shi, Japan, 9–12 June 2019; pp. 901–906.
18. Li, L. Time-of-Flight Camera—An Introduction, Texas Instruments, May, 2014. Available online: https://www.ti.com/product/OPT8241#tech-docs (accessed on 3 April 2022).
19. Giancola, S.; Valenti, M.; Sala, R. *A Survey on 3D Cameras: Metrological Comparison of Time-Of-Flight, Structured-Light and Active Stereoscopy Technologies*; Springer: Berlin/Heidelberg, Germany, 2018.
20. Hansard, M.; Lee, S.; Choi, O.; Horaud, R. *Time of Flight Cameras: Principles, Methods, and Applications*; Springer Science & Business Media: New York, NY, USA, 2012.
21. Park, H.; Van Messem, A.; De Neve, W. Box-Scan: An Efficient and Effective Algorithm for Box Dimension Measurement in Conveyor Systems using a Single RGB-D Camera. In Proceedings of the the 7th IIAE International Conference on Industrial Application Engineering, Kitakyushu, Japan, 26–30 March 2019. [CrossRef]
22. Leo, M.; Natale, A.; Del-Coco, M.; Carcagni, P.; Distante, C. Robust Estimation of Object Dimensions and External Defect Detection with a Low-Cost Sensor. *J. Nondestruct. Eval.* **2017**, *36*, 17. [CrossRef]
23. Ferreira, B.; Griné, M.; Gameiro, D.; Costeira, J.; Santos, B. VOLUMNECT: Measuring volumes with Kinect. In Proceedings of the SPIE—The International Society for Optical Engineering, San Francisco, CA, USA, 6 March 2014. [CrossRef]
24. Fischler, M.; Bolles, R. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* **1981**, *24*, 381–395. [CrossRef]
25. Vezzani, G.; Pattacini, U.; Natale, L. A grasping approach based on superquadric models. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 1579–1586.

26. Makhal, A.; Thomas, F.; Gracia, A.P. Grasping unknown objects in clutter by superquadric representation. In Proceedings of the 2nd IEEE International Conference on Robotic Computing (IRC), Laguna Hills, CA, USA, 31 January–2 February 2018; pp. 292–299.
27. Vezzani, G.; Pattacini, U.; Pasquale, G.; Natale, L. Improving Superquadric Modeling and Grasping with Prior on Object Shapes. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 6875–6882.
28. Haschke, R.; Walck, G.; Ritter, H. Geometry-Based Grasping Pipeline for Bi-Modal Pick and Place. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 27 September–1 October 2021; pp. 4002–4008.
29. Tomašević, D.; Peer, P.; Solina, F.; Jaklič, A.; Štruc, V. Reconstructing Superquadrics from Intensity and Color Images. *Sensors* **2022**, *22*, 5332. [CrossRef] [PubMed]
30. Solina, F.; Bajcsy, R. Range image interpretation of mail pieces with superquadrics. In Proceedings of the National Conference on Artificial Intelligence, Seattle, WA, USA, 13–17 July 1987; Volume 2, pp. 733–737.
31. Jaklič, A.; Erič, M.; Mihajlović, I.; Stopinšek, Ž.; Solina, F. Volumetric models from 3D point clouds: The case study of sarcophagi cargo from a 2nd/3rd century AD Roman shipwreck near Sutivan on island Brač, Croatia. *J. Archaeol. Sci.* **2015**, *62*, 143–152. [CrossRef]
32. Quispe, A.H.; Milville, B.; Gutiérrez, M.A.; Erdogan, C.; Stilman, M.; Christensen, H.; Amor, H.B. Exploiting symmetries and extrusions for grasping household objects. In Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 26–30 May 2015; pp. 3702–3708. [CrossRef]
33. Mitash, C.; Wang, F.; Lu, S.; Terhuja, V.; Garaas, T.; Polido, F.; Nambi, M. ARMBench: An object-centric benchmark dataset for robotic manipulation. *arXiv* **2023**, arXiv:2303.16382.
34. Burke, C.; Nguyen, H.; Magilligan, M.; Noorani, R. Study of A Drone's Payload Delivery Capabilities Utilizing Rotational Movement. In Proceedings of the 2019 International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST), Dhaka, Bangladesh, 10–12 January 2019; pp. 672–675. [CrossRef]
35. Colling, D.; Dziedzitz, J.; Furmans, K.; Hopfgarten, P.; Markert, K. Progress in Autonomous Picking as Demonstrated by the Amazon Robotic Challenge. In Proceedings of the 15th IMHRC, Savannah, GA, USA, 23–26 July 2018.
36. Zeng, A.; Song, S.; Yu, K.T.; Donlon, E.; Hogan, F.R.; Bauza, M.; Ma, D.; Taylor, O.; Liu, M.; Romo, E.; et al. Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. *Int. J. Robot. Res.* **2022**, *41*, 690–705. [CrossRef]
37. Bottarel, F.; Vezzani, G.; Pattacini, U.; Natale, L. GRASPA 1.0: GRASPA is a Robot Arm graSping Performance BenchmArk. *IEEE Robot. Autom. Lett.* **2020**, *5*, 836–843. [CrossRef]
38. Garrido-Jurado, S.; Muñoz-Salinas, R.; Madrid-Cuevas, F.J.; Marín-Jiménez, M.J. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognit.* **2014**, *47*, 2280–2292. [CrossRef]
39. Open Source Computer Vision—Detection of ArUco Markers, OpenCV. Available online: https://docs.opencv.org/trunk/d5/dae/tutorial_aruco_detection.html (accessed on 11 May 2016).
40. Barr, A.H. Superquadrics and angle-preserving transformations. *IEEE Comput. Graph. Appl.* **1981**, *1*, 11–23. [CrossRef]
41. Biederman, I. Human image understanding: Recent research and a theory. *Comput. Vis. Graph. Image Process.* **1985**, *32*, 29–73. [CrossRef]
42. Solina, F.; Ruzena, B. Recovery of parametric models from range images: The case for superquadrics with global deformations. *IEEE Trans. Pattern Anal. Mach. Intell.* **1990**, *12*, 131–147. [CrossRef]
43. Jaklic, A.; Leonardis, A.; Solina, F. *Segmentation and Recovery of Superquadrics*; Kluwer Academic Publishers: Dordrecht, The Netherlands, 2000.
44. Roweis, S. *Topic: "Levenberg-Marquardt Optimization"*; University of Toronto: Toronto, ON, Canada, 1996.
45. OPT8241—QVGA-Resolution 3D Time-of-Flight (ToF) Sensor, Texas Instruments. Available online: https://www.ti.com/lit/ds/symlink/opt8241.pdf?ts=1698199280137&ref_url=https%253A%252F%252Fwww.google.com%252F (accessed on 11 May 2016).
46. Texas Instruments, Voxel Viewer User's Guide. Available online: https://www.ti.com/lit/ug/sbou157/sbou157.pdf (accessed on 11 May 2016).
47. *MATLAB*, Version 9.9.0. 1467703 (R2020b); The MathWorks Inc.: Natick, MA, USA, 2020.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

sensors

MDPI

Article

Efficient Large-Scale Point Cloud Geometry Compression

Shiyu Lu [1,2,*], Cheng Han [2,*] and Huamin Yang [2,*]

[1] School of Computer Science and Technology, Changchun University, Changchun 130022, China
[2] School of Computer Science and Technology, Changchun University of Science and Technology, Changchun 130022, China
* Correspondence: lusy@ccu.edu.cn (S.L.); hancheng@cust.edu.cn (C.H.); yanghuamin@cust.edu.cn (H.Y.)

Abstract: Due to the significant bandwidth and memory requirements for transmitting and storing large-scale point clouds, considerable progress has been made in recent years in the field of large-scale point cloud geometry compression. However, challenges remain, including suboptimal compression performance and complex encoding–decoding processes. To address these issues, we propose an efficient large-scale scene point cloud geometry compression algorithm. By analyzing the sparsity of large-scale point clouds and the impact of scale on feature extraction, we design a cross-attention module in the encoder to enhance the extracted features by incorporating positional information. During decoding, we introduce an efficient generation module that improves decoding quality without increasing decoding time. Experiments on three public datasets demonstrate that, compared to the state-of-the-art G-PCC v23, our method achieves an average bitrate reduction of −46.64%, the fastest decoding time, and a minimal network model size of 2.8 M.

Keywords: point cloud geometry compression; cross-attention; efficient generation

1. Introduction

In the real world, most space is unoccupied by observed objects. Large-scale scene point clouds effectively reduce data volume while preserving spatial structure information, capturing the properties of large-scale scenes. This makes them well-suited for representing the 3D structure of expansive environments, leading to widespread applications in fields such as autonomous driving [1], robotics [2], and virtual reality [3]. However, in practical applications, point cloud data can contain hundreds of millions of points or even reach TB-level sizes. This not only requires substantial storage space but also presents significant challenges in data processing, management, sharing, and application, thereby raising higher demands for point cloud compression. The Moving Picture Experts Group (MPEG) [4] has proposed two traditional point cloud compression standards: video-based V-PCC and geometry-based G-PCC. V-PCC first converts point cloud sequences into video format and applies traditional video compression techniques, while G-PCC uses an efficient octree structure to eliminate redundant information and achieve compression. However, these traditional methods rely on handcrafted features and are limited in their applicability. With the growing use of deep learning in image and video compression, its application to large-scale scene point cloud compression is also gaining increasing attention.

In recent years, deep learning has made significant progress in point cloud compression. Many studies have focused on dense objects or human point clouds. Refs. [5–8] proposed a series of lossy geometric compression algorithms for point clouds, achieving high reconstruction quality at low bitrates. Refs. [9–12] introduced a set of lossless geometric compression algorithms, ensuring low bitrates while maintaining lossless quality.

https://doi.org/10.3390/s25051325

However, these algorithms are less effective for large-scale scene point clouds, where the sparse distribution of points makes compression more challenging. This sparsity results in numerous empty voxels during the voxelization of sparse point clouds, significantly hindering subsequent processing. Currently, voxel-based methods for large-scale scene point clouds [13–16] convert them into octree structures to reduce computational and parameter requirements. However, these methods still demand substantial computation, limiting their applicability. With the advent of efficient models like PointNet [17] and PointNet++ [18], it has become feasible to process point cloud data directly and efficiently. Thus, our approach to the geometric compression of large-scale scene point clouds leverages direct point-based processing. However, due to the high complexity and large scale of such point clouds, existing methods suffer from suboptimal compression performance and complex decoding processes [19,20]. Recent studies [21] have attempted to address these issues by designing an attention-based encoder to enhance compression performance for sparse point clouds and by developing a folding-based point cloud generation module to reduce decoding time. However, the attention module in these designs fails to fully leverage the positional information introduced and does not effectively recover high-dimensional geometric features, resulting in subpar decoding quality. To overcome this, we designed a cross-attention module and an optimized generation module, achieving both lower decoding time and improved decoding quality.

In summary, the main innovations of this work are as follows:

- We propose an efficient large-scale scene point cloud geometry compression algorithm with a network model size of only 2.8 M, making it suitable for mobile applications.
- We design a cross-attention module that deeply integrates positional and feature information, enhancing feature extraction quality and improving compression performance.
- We develop an optimized generation module that effectively recovers high-dimensional geometric features, enhancing decoding quality without increasing decoding time.
- Extensive comparative and ablation experiments demonstrate that the proposed method achieves state-of-the-art performance across three datasets and delivers superior results in terms of subjective quality as well.

2. Related Work

In recent years, substantial progress has been made in large-scale scene point cloud compression, driving rapid advancements in the field. Broadly speaking, existing research can be categorized into two main types: lossy and lossless compression for large-scale scene point clouds. The following is a review of relevant work on geometric information compression for large-scale scene point clouds.

2.1. Lossy Geometry Compression for Large-Scale Point Clouds

For lossy large-scale scene point cloud compression, Huang et al. [19] proposed a deep learning-based point cloud compression network that effectively processes various types of point clouds by learning common structural features, including simple and sparse shapes. Wiesmann et al. [20] introduced an innovative convolutional autoencoder architecture that operates directly on the points themselves, thus avoiding voxelization; however, the reconstruction quality remains suboptimal. Liang et al. [22] employed a Transformer-based encoder architecture for point cloud geometry compression, where the input point cloud is treated as a continuous spatial set with learnable positional embeddings and compressed using self-attention layers and point-wise operations. Pang et al. [23] used multilayer perceptrons and convolutional neural networks as the backbone, giving their approach an advantage in handling sparse point clouds and enabling post-processing techniques for further refinement of the decompressed point clouds. Later, Pang et al. [24]

proposed a geometry compression algorithm for point clouds that supports point, voxel, and tree representations, featuring a context-aware up-sampling module for decoding and an enhanced voxel Transformer module for feature aggregation. Cui et al. [14] used non-overlapping context windows to construct sequences and shared the results of the multi-head self-attention mechanism, reducing time overhead.

For large-scale point cloud scenes, Sun et al. [25] converted discrete voxels into fine-grained 3D points, effectively addressing the coordinate loss that occurs during the octree generation process. Fan et al. [26] explored the spatial correlation across different layers through progressive down-sampling, modeling the corresponding residuals with a fully decomposed entropy model, thereby achieving compression of latent variables. Inspired by IPDAE [7], Huang et al. [27] proposed an ordered segmentation algorithm based on patch-wise point cloud compression, tailored to the specific characteristics of point clouds. Wang et al. [28] decoupled the original point cloud into multiple layers of point subsets, compressing and transmitting each layer independently to ensure reconstruction quality requirements across different scenarios. You et al. [21] proposed an attention-based encoder that embeds features from local windows and introduces dilated windows as cross-scale priors to infer the distribution of quantized features in parallel.

2.2. Lossless Geometry Compression for Large-Scale Point Clouds

For lossless large-scale scene point cloud compression, Huang et al. [13] utilized the sparsity and structural redundancy between points to reduce bitrate. Biswas et al. [29] modeled the probabilities of octree symbols and associated intensity values by exploiting temporal and spatial correlations in the data. However, this work overlooked neighboring nodes and local geometric features in the current point cloud frame. Que et al. [15] proposed a two-stage deep learning framework that combines the strengths of octree-based and voxel-based methods, using voxel context to compress octree-structured data. Fu et al. [16] first employed an octree representation to reduce spatial redundancy, making it robust to point clouds at different resolutions. They then used a conditional entropy model with a large receptive field to model sibling and ancestor contexts, calculating strong dependencies between neighboring nodes, and applied an attention mechanism to emphasize relevant nodes in the context. This extensive context includes ancestor nodes, thousands of neighboring nodes, and their ancestor nodes, covering nearly the entire octree. However, such heavy global attention computations and autoregressive context are inefficient for practical applications.

Building on OctAttention [16], Song et al. [30] introduced a hierarchical attention structure with linear complexity for context scales, preserving a global receptive field. Additionally, they designed a grouped context structure to address the serial decoding issue caused by autoregression while maintaining compression performance. However, since these studies primarily focus on optimizing network structure and context information, they often overlook fundamental training strategies and efficient context utilization. Song et al. [31] later developed a geometry-aware feature extraction module capable of extracting effective features from large-scale contexts. Lodhi et al. [32] employed sparse 3D convolutions to extract features across various octree scales for lossless compression of point cloud octree representations. Luo et al. [33] transformed sparse point clouds from Cartesian to spherical coordinates, simplifying redundancy reduction for neural networks. By applying the spherical coordinate system to Cartesian-based methods like EHEM [30] and OctAttention [16], they demonstrated its effectiveness.

3. Proposed Method

The network architecture of the efficient large-scale scene point cloud geometry compression algorithm is shown in Figure 1. This network consists of an encoder, entropy coding module, decoder, and a loss function. During encoding and decoding, the encoder first extracts features from the input point cloud. The Octree-predlift method from G-PCC is then used to encode and decode the down-sampled key points. The entropy coding module estimates the distribution of the geometric features extracted by the encoder and encodes them into a binary file. Finally, the decoder reconstructs the sparse point cloud from the decoded geometric features and key points.

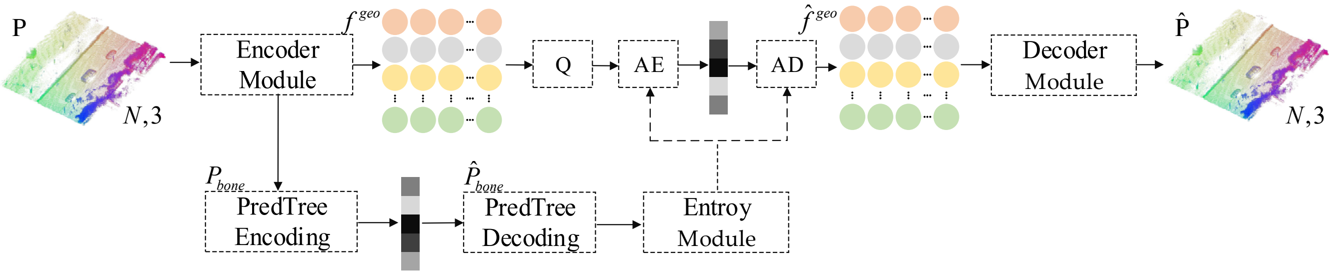


Figure 1. Network architecture overview.

3.1. Encoder Module

The encoder is mainly composed of keypoint sampling, window querying, adaptive alignment, and cross-attention modules, as shown in Figure 2. The sampling and querying methods in the proposed algorithm use aggregation operations commonly applied in point cloud analysis tasks, such as farthest point sampling [34], random point sampling [35], and K-nearest neighbor [36]. The encoder module is described in detail below.

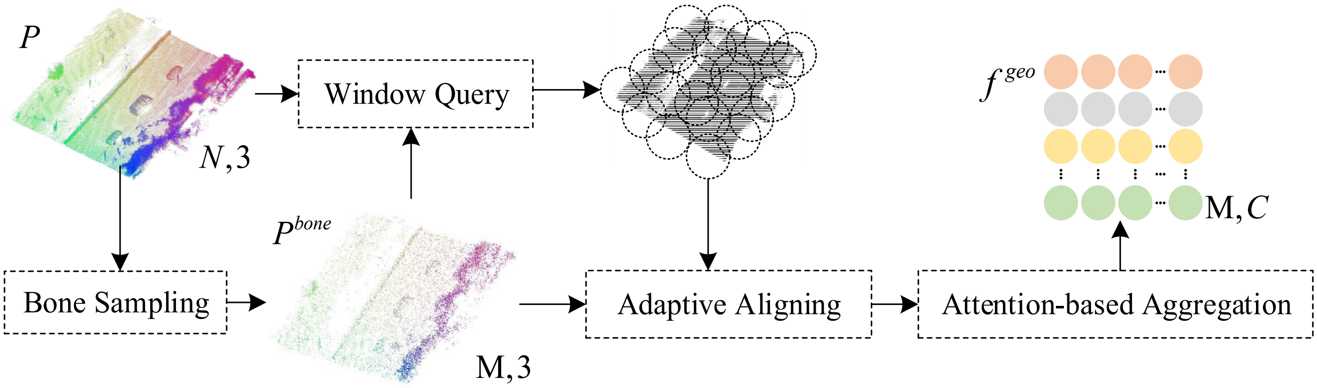


Figure 2. Encoder module.

Due to the high computational cost of farthest point sampling, the original point cloud $P \in R^{N\times 3}$ is first randomly down-sampled to obtain a subset with no more than $M \times 16$ points. Farthest point sampling is then applied to this subset to obtain $P^{bone} \in R^{M\times 3}$ with M points, where P^{bone} represents the key points. Next, adaptive alignment is performed on overlapping local windows, moving each window to the coordinate origin, and finally, rescaling is conducted based on the key points, as follows:

$$d = \frac{1}{|P^{bone}|} \sum_{p_i, p_j \in P^{bone}} \min\{||p_i - p_j||_2 : p_i \neq p_j\}, \tag{1}$$

$$P_i^W = \left\{ \frac{p - p_i}{d} : p \in T(p_i, P, K) \right\}, \forall p_i \in P^{bone}. \tag{2}$$

where P_i^W is the aligned local window, $T(p_i, P, K)$ denotes the K nearest neighbors of point p_i found within the input point cloud P, and d is the scaling factor calculated from P^{bone}.

3.1.1. Cross-Attention Module

To effectively extract feature information from the local window $P_i^W \in R^{K\times 3}$, a cross-attention module was designed, as shown in Figure 3. Positional information is introduced to enhance the features, after which the extracted features are concatenated and restored to (K, C) through a multi-layer perceptron. Finally, high-dimensional feature vector $F_i^{geo} \in R^{1\times C}$ is extracted using max-pooling and then undergoes entropy encoding.

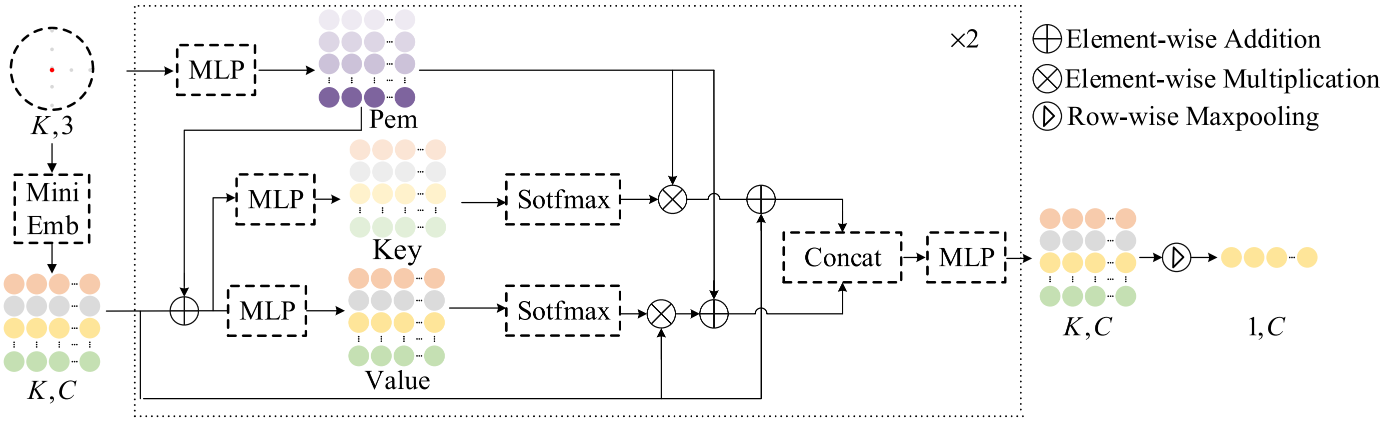


Figure 3. Cross-attention module.

Specifically, an embedding operation based on graph convolution is first performed on each point within the local window to generate the feature $F_i^{(0)} \in R^{K\times C}$ that captures local details, as follows:

$$F_i^{(0)}[j] = GraphConv(T(p_j^i, P_i^{AW}, K)), \forall p_j^i \in P_i^{AW}. \tag{3}$$

where $T(p_j^i, P_i^{AW}, K)$ represents the K nearest neighbors of point p_j found within the aligned window P_i^{AW}, $GraphConv$ is defined as $GraphConv(\cdot) = MaxPool(MLP(\cdot))$, and $F_i^{(0)}[j] \in R^{1\times C}$ denotes the j-th feature vector in the feature matrix $F_i^{(0)} \in R^{K\times C}$, corresponding to point p_j^i. Next, a stacked cross-attention module is executed, with the process of the l-th attention block described as follows:

$$Pem_i^{(l)} = MLP(position), \tag{4}$$

$$Key_i^{(l)} = \sigma(MLP(F_i^{(l)} + Pem_i^{(l)}) \times F_i^{(l)} + Pem_i^{(l)}), \tag{5}$$

$$Value_i^{(l)} = \sigma(MLP(F_i^{(l)} + Pem_i^{(l)}) \times Pem_i^{(l)} + F_i^{(l)}), \tag{6}$$

$$F_i^{(l+1)} = MLP(Concat(Key_i^{(l)}, Value_i^{(l)})). \tag{7}$$

where σ represents the Softmax operation, Pem denotes the positional encoding multiplier, and Key and $Value$ refer to the feature information extracted through an MLP. Finally, the features $F_i^{(l)} \in R^{K\times C}$ output from the last cross-attention module are aggregated into the geometric feature $F_i^{geo} \in R^{1\times C}$ using max-pooling, as follows:

$$F_i^{geo} = MaxPool(F_i^{(L)}). \tag{8}$$

3.1.2. Dilation Window Down-Sampling Module

Since there is dependency between the target local window and the neighboring area of the down-sampled key points, a dilated window is used as a cross-scale prior before entropy encoding. This approach captures the neighboring relationships within the point cloud in the local window, enabling the model to consider both local detail features and the point cloud distribution information over a wider area. Additionally, to achieve fast arithmetic encoding, the geometric features extracted by the encoder are further compressed using graph convolution and fully connected layers. Specifically, as the key points are encoded by G-PCC, the dilated neighborhood can serve as a cross-scale prior. The dilated window P_i^{DW} is defined as follows:

$$P_i^{DW} = T(p_i, \hat{P}^{bone}, K), \forall p_i \in \hat{P}^{bone}. \tag{9}$$

where $P_i^{AW} \in R^{k\times 3}$, T represents the K-nearest neighbors of the down-sampled point p_i in the down-sampled $\hat{P}^{bone}$, with K set to 8 in the experiment.

Due to the longer symbol sequences and higher-dimensional features to be encoded, the complexity of arithmetic encoding increases. Therefore, a simple fully connected layer is used to perform the compression operation, defined as follows:

$$f_i^{geo} = Linear(F_i^{geo}). \tag{10}$$

where $F_i^{geo} \in R^{1\times C}$ and $f_i^{geo} \in R^{1\times c}$ are the geometric features, with $C = 128$ and $c = 16$ set in the experiment. After arithmetic decoding, a linear layer is also needed to restore the geometric features back to F_i^{geo}.

3.2. Entropy Module

In the entropy encoding module, uniform quantization is employed, which is replaced with added uniform noise during training, resulting in the quantized geometric features denoted as $\widetilde{f}^{geo} = Q(f^{geo})$, as follows:

$$P_\theta(\widetilde{f}^{geo}) = \textstyle\prod_{i=1}^{M} (L(\Phi^i) \times U(-1/2, 1/2))(\widetilde{f}_i^{geo}). \tag{11}$$

where P_τ represents the entropy model parameterized by τ, $L(\Phi_i)$ refers to the Laplace distribution of the quantized features $\widetilde{f}^{geo}$, and the parameters $\Phi_i = (\mu_i, \sigma_i)$ and $U(-1/2, 1/2)$ denote a uniform distribution over the interval $[-1/2, 1/2]$. The parameter Φ_i can be estimated from the dilated window using a network that includes a *GraphConv* layer and an *MLP*, as follows:

$$\Phi_i = (\mu_i, \sigma_i) = MLP(GraphConv(P_i^{DW})). \tag{12}$$

Finally, the bit rate of the geometric features is calculated as follows:

$$R^{geo} = -\frac{1}{N} \log_2 P_\tau(\widetilde{f}^{geo}). \tag{13}$$

where N represents the number of points in the input point cloud.

3.3. Decoder Module

Due to the significant complexity that multi-scale methods may introduce to the decoder, our approach employs a single-scale strategy to reduce computational demands during decoding. Since the number of geometric features M at the key points is much smaller than the original number of input points N, we first process the geometric features using a feature refinement module. Then, we utilize the designed efficient generation module to generate the position information of the points. Finally, the reconstruction of the point cloud is restored to its original position using a backward alignment operation, as shown in Figure 4.

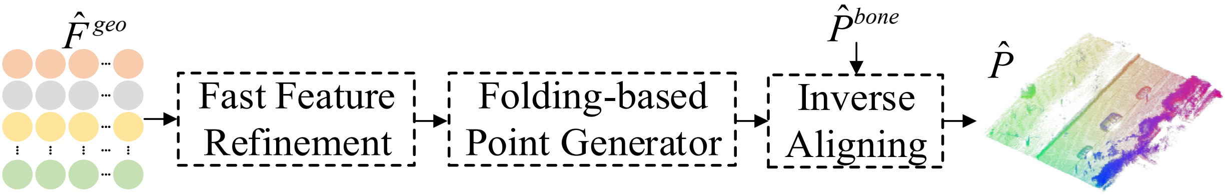


Figure 4. Decoder module.

The feature refinement module consists primarily of a dilated window convolution (DW-Conv) and a linear layer, as shown in Figure 5a. The DW-Conv integrates information from the dilated window, reducing redundant computations in the spatial graph and enabling a graph structure for feature convolution. Specifically, using the provided dilated indices, the features of the points within the corresponding dilated window are grouped and collected. Graph convolution (Graph-Conv) is then applied to aggregate these collected groups, resulting in refined features, as illustrated in Figure 5b.

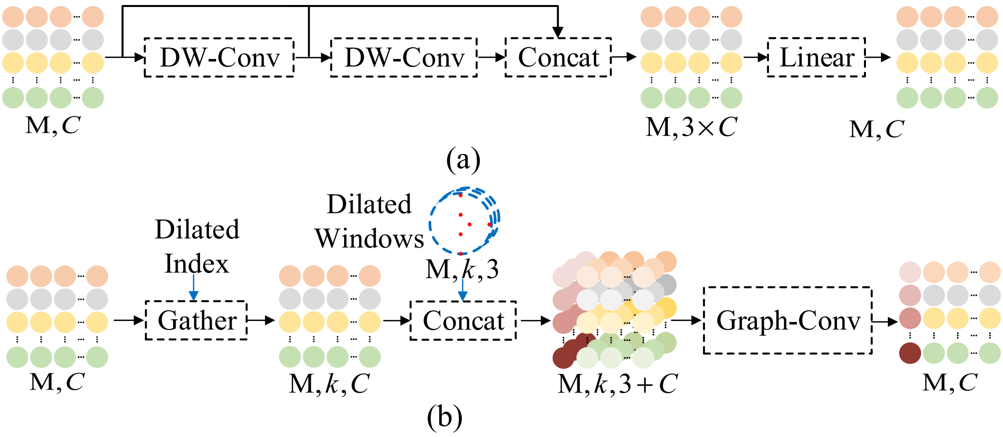


Figure 5. Feature refinement module: (**a**) feature refinement; (**b**) dilated window convolution.

The geometric features processed by the feature refinement module are transformed into point coordinate information using the designed efficient generation module, as shown in Figure 6. Here, an *MLP* is used for up-sampling the input features, and a *Reshape* operation adjusts the output dimensions. The efficient generation module employs a dual-branch structure to extract features. The first branch converts the geometric features from a $1 \times C$-dimensional vector into a $R_{\max} \times D$ grid matrix, which is then randomly sampled to produce $R \times D$. The second branch also converts the geometric features into $R \times D$ using *Reshape* and *MLP*. Finally, the up-sampled geometric features from both branches are concatenated and passed through an *MLP* and *Reshape* operation to generate the point coordinates.

The reverse alignment operation is the inverse process of the adaptive alignment operation used in the encoder. Each reconstructed window $\hat{P}_i^{AW}$ is moved back to its original position and restored to its original scale to assemble into the complete reconstruction result $\hat{P}$, as follows:

$$\hat{P} = \cup_{\hat{p}_i \in \hat{P}^{bone}} \left\{ (\hat{p} \times \hat{d}) + \hat{p}_i : \hat{p} \in \hat{P}_i^{AW} \right\}. \tag{14}$$

where $\hat{d}$ is the scaling factor recalculated by $\hat{P}^{bone}$.

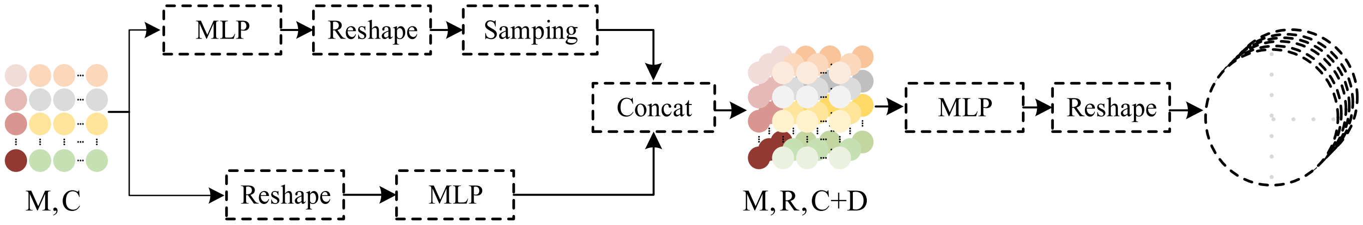


Figure 6. Efficient generation module.

3.4. Loss Function

The chamfer distance [37] is chosen as the loss function to measure the average distance from each point in one point set to its nearest neighbor in the other point set, as follows:

$$D_{CD}(P,\hat{P}) = \vec{d}^2(P,\hat{P}) + \vec{d}^2(\hat{P},P). \tag{15}$$

where D_{CD} represents the chamfer distance loss function, P denotes the input point cloud, $\hat{P}$ signifies the decompressed point cloud, and $\vec{d}^2$ is the average symmetric squared distance from the input point cloud to the nearest neighbors in the decompressed point cloud.

The total loss function L_{total} is constructed by adding D_{CD} and the geometric feature bit rate R_{geo}, as follows:

$$L_{total} = D_{CD} + \lambda R_{geo}. \tag{16}$$

where the calculation of R_{geo} is shown in (13), where λ is the loss weight used to balance the impact of multi-scale loss, optimizing the parameters of the entire network within the end-to-end training scheme.

4. Experimental Results

The large-scale point cloud geometry compression algorithm is trained on the ShapeNet dataset [38], which contains 35,708 point clouds, each generated by uniformly sampling 8000 points from CAD models. The test dataset includes both indoor and outdoor scenes: the indoor point clouds are sourced from S3DIS [39] and ScanNet [40], while the outdoor point cloud map is derived from KITTI [41], processed according to ref. [20]. Detailed information about the test set is provided in Table 1. Our method is implemented using Python 3.10 and PyTorch 2.0 for network model training. The experimental setup includes the following parameters: the optimizer is Adam [42] with an initial learning rate of 0.0005, a batch size of 1, and the model is trained for 140,000 iterations with a local window size of 128. The bitrate–distortion balance is set to 10^{-4}, and G-PCC [4] is used for lossless compression of the down-sampled key points. All experiments were conducted on an AMD Ryzen 7 5800X CPU (AMD, Sunnyvale, CA, USA) and an NVIDIA RTX 2080Ti GPU (NVIDIA, Santa Clara, CA, USA).

Table 1. Test dataset details.

Data	Test	Model Number	Point Number
KITTI	Area 6	48	554 K~214 K
S3DIS	Official test set	100	3.2 M~0.3 M
ScanNet	Sequence 08	186	553 K~32 K

4.1. Evaluation Indicators

In the image domain, objective metrics such as symmetric nearest neighbor, root mean square error, and peak signal-to-noise ratio (PSNR) are commonly used to assess the quality of decoded images. However, point clouds not only involve errors in color attribute information but also errors in geometric data compression. Therefore, it is necessary to evaluate attribute distortion and geometric distortion separately. To assess the data fidelity of compression algorithms, the geometric distortion levels of decoded point clouds at different bitrates are typically compared, a process referred to as bitrate–distortion optimization. In this context, geometric distortion generally refers to the geometric error between the decompressed point cloud and the original point cloud, while bitrate refers to the number of bits required to encode each input point (bpp).

$$\text{bpp} = \frac{S}{N}. \tag{17}$$

where S represents the size of the encoded data file in bits, and N denotes the number of points in the original point cloud.

Geometric distortion in point cloud data is typically measured by the distance between the points in the original point cloud and the points in the decoded point cloud. To calculate the distance between a point in the original point cloud and its corresponding point in the decoded point cloud, the correspondence between the original and decoded point clouds must first be established. Then, the distance deviation between corresponding points is used to compute objective evaluation metrics such as geometric peak signal-to-noise ratio. For both the original and decoded point clouds, distortion is calculated separately. A matching relationship is established for the reference point cloud, and based on the corresponding points, coordinate distortion or color distortion is computed. The larger distortion value is taken as the symmetric distortion, commonly referred to as the D1 metric. To calculate the distance from a point to a plane, surface reconstruction of the point cloud is performed first, and then the point-to-surface distance is used in place of the point-to-point distance for the calculation, which is typically called the D2 metric.

The D1 metric is defined by connecting a point a_j in the original point cloud A to a point b_i in the decoded point cloud B to determine the error vector $E(i,j)$. The calculation formula is as follows:

$$e_{B,A}^{D1}(i) = \left|\left|E(i,j)\right|\right|_2^2. \tag{18}$$

where $E(i,j)$ represents the point-to-point error. Based on the distance from all points $(i \in B)$ in the decoded point cloud B to the corresponding points in the original point cloud, the distance is denoted as $e_{B,A}^{D1}$. N_B represents the total number of points in the decoded point cloud B. The D1 metric can thus be expressed as follows:

$$\varepsilon_{B,A}^{D1} = \max_{b_i \in B}\left\{e_{B,A}^{D1}(i)\right\}. \tag{19}$$

The D2 metric is calculated by projecting the error vector $E(i,j)$ along the normal vector N_j to obtain a new error vector $E(i,j)$. This projection accounts for the distance from a point to the plane. The calculation formula for the point-to-plane error is as follows:

$$e_{B,A}^{D2}(i) = \left|\left|E(i,j)\right|\right|_2^2 = (E(i,j) \cdot N_j)^2. \tag{20}$$

To assess the distortion, the peak signal-to-noise ratio (PSNR) is used. The geometric PSNR is calculated based on the geometric errors D1 and D2, and the formula for calculating the geometric PSNR is as follows:

$$PSNR = 10\log_{10}\left(\frac{p^2}{\max(e_{B,A}^{D_x}, e_{A,B}^{D_x})}\right). \tag{21}$$

where p represents the geometric peak of the decoded point cloud.

4.2. Performance Evaluation

The proposed method is compared with the state-of-the-art traditional method, G-PCCv23, where G-PCCv23(O) serves as the default octree-based encoder–decoder. Additionally, a comparative analysis is performed with deep learning-based methods, including OctAttention, IPDAE, and Pointsoup. To ensure a fair comparison, all deep learning-based methods were retrained on the same dataset as our method, and all test samples were normalized to the coordinate range [0, 1023].

To evaluate distortion in lossy compression, we perform an objective comparison using rate–distortion optimization, deriving D1-PSNR from point-to-point error, with bitrate measured in bits per point. Since the test datasets lack reference normals, which are required for calculating D2 PSNR, comparisons for D2 PSNR are not included. Figure 7 shows the rate–distortion optimization curves of our method on the S3DIS, ScanNet, and KITTI datasets, where the proposed method clearly outperforms others in D1-PSNR across all three datasets. Table 2 presents the bitrate gains in D1-PSNR compared to G-PCCv23(O), OctAttention, IPDAE, and Pointsoup, with negative values indicating bitrate savings (i.e., better compression performance). As shown, OctAttention and IPDAE achieve reduced average bitrates across all three datasets compared to G-PCCv23(O), reflecting inferior compression performance. Pointsoup achieves a −44.66% bitrate gain on average in terms of D1-PSNR compared to G-PCCv23(O), while our method achieves a −46.62% bitrate gain, delivering the best experimental results. Table 3 compares the encoding and decoding times of different methods, showing that G-PCCv23(O) has the fastest encoding time, while our method achieves the fastest decoding time. The real-time decoding algorithm we propose is well-suited for tasks such as map construction and updating in dynamic environments. The longer encoding time of our method is attributed to the use of a more complex encoder; however, we plan to design a more efficient encoder module to enable real-time encoding performance. Additionally, as shown in Table 4, our method has the smallest model parameter size of 2.8 M, making it more lightweight compared to G-PCCv2.3 and Pointsoup.

Table 2. Bit rate gain of different methods compared with G-PCCv23(O) on D1 PSNR.

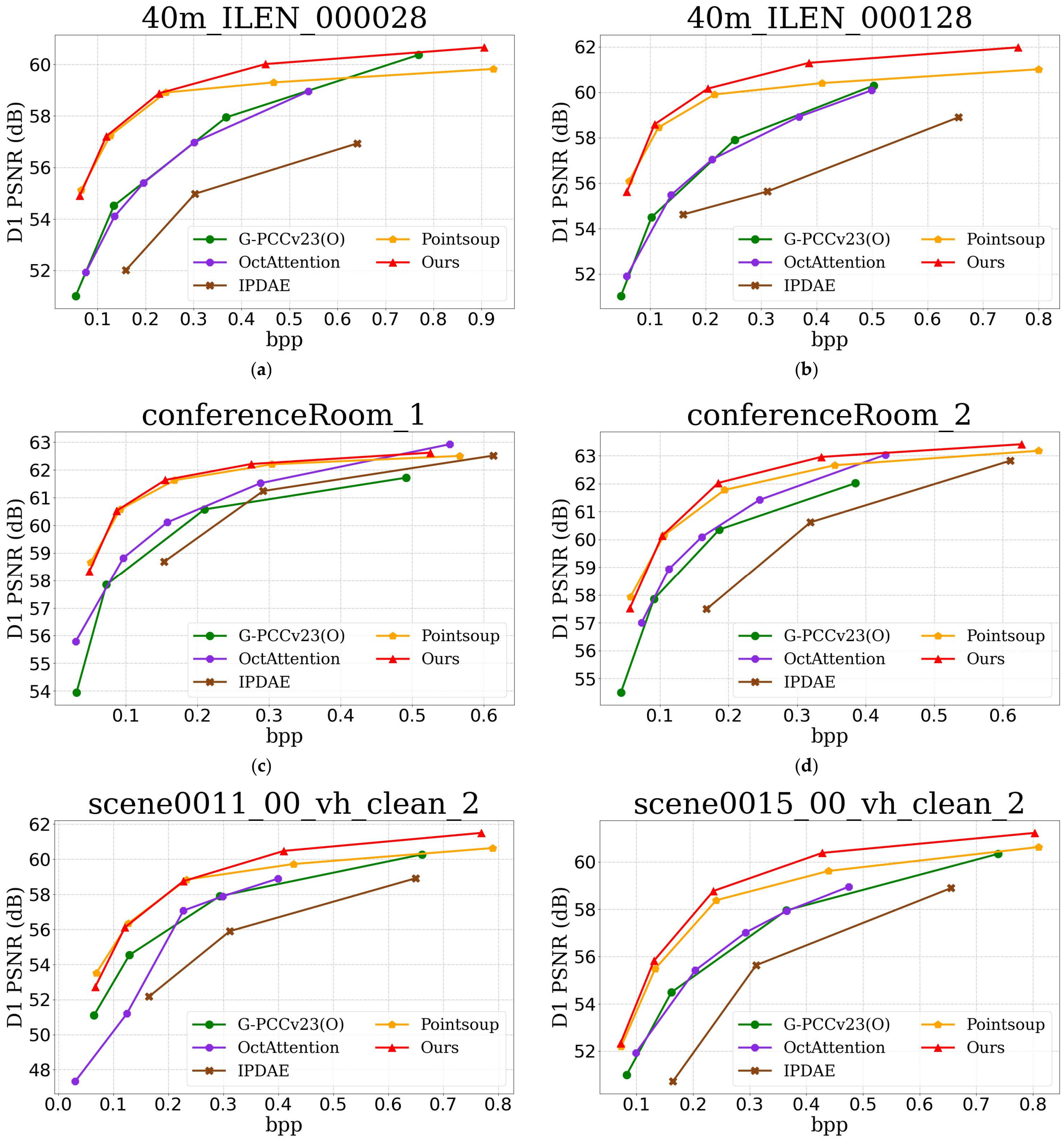
Dataset	OctAttention	IPDAE	Pointsoup	Ours
S3DIS	−13.90%	+15.29%	−51.07%	**−51.16%**
ScanNet	+28.34%	+80.80%	−33.35%	**−34.58%**
KITTI	+11.6%	+108.23%	−49.57%	**−54.12%**
Avg.	+8.68%	+68.11%	−44.66%	**−46.62%**

Table 3. Comparison of encoding and decoding time of different methods.

	G-PCCv23(O)	OctAttention	IPDAE	Pointsoup	Ours
Times/s	Enc/Dec	Enc/Dec	Enc/Dec	Enc/Dec	Enc/Dec
S3DIS	**0.33**/0.13	0.55/386.60	20.85/1.22	6.96/**0.06**	7.16/**0.06**
ScanNet	**0.06**/0.03	0.15/59.50	4.64/0.24	1.77/**0.02**	1.74/**0.02**
KITTI	**0.11**/0.05	0.17/62.81	6.94/0.46	1.91/0.04	1.97/**0.03**
Avg.	**0.17**/0.07	0.29/169.64	10.81/0.64	3.55/**0.04**	3.62/**0.04**

Table 4. Comparison of model size of different methods.

	G-PCCv23(O)	OctAttention	IPDAE	Pointsoup	Ours
Model Size	5.3 MB	28.0 MB	18.8 MB	2.9 MB	**2.8 MB**



(**a**)

(**b**)

(**c**)

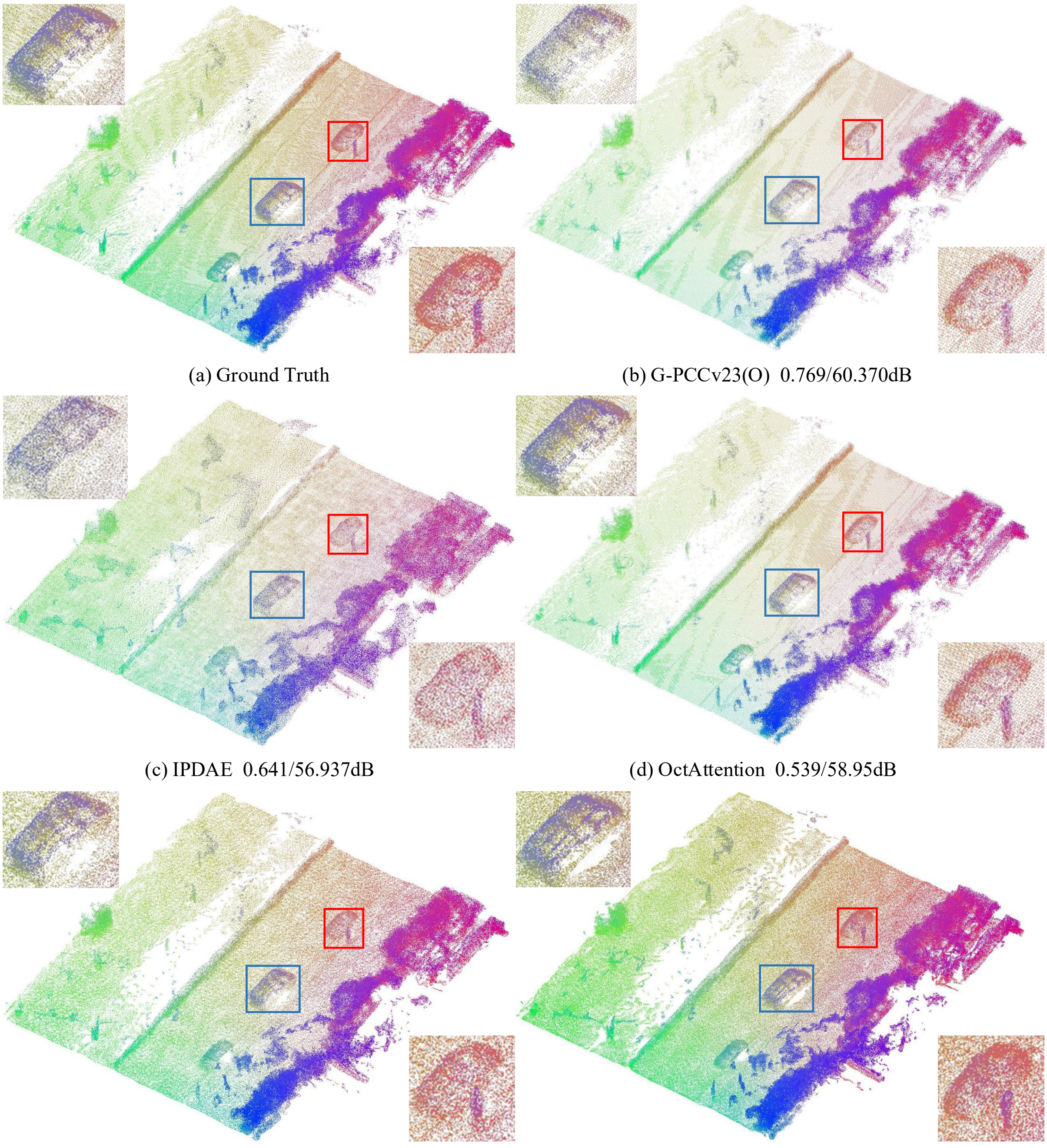
(**d**)

(**e**)

(**f**)

Figure 7. Comparison of bitrate–distortion optimization curves of different methods. (**a**) is the D1 curve of the 40 m_ILEN_000028 point cloud in S3DIS. (**b**) is the D1 curve of the 40 m_ILEN_000128 point cloud in S3DIS. (**c**) is the D1 curve of the conferenceRoom_1 point cloud in ScanNet. (**d**) is the D1 curve of the conferenceRoom_2 point cloud in ScanNet. (**e**) is the D1 curve of the scene0011_00_vh_clean_2 point cloud in KITTI. (**f**) is the D1 curve of the scene0015_00_vh_clean_2 point cloud in KITTI.

Figures 8–10 provide a visual comparison of the point clouds decoded by all methods for both indoor and outdoor scenes. To enhance the visual quality of large-scale point clouds, color rendering is applied, and detailed zoom-in views are provided, indicated by the red and blue boxes in the figures. In Figure 8, the blue zoom-in detail shows that our reconstruction most closely matches the vehicle contours in the original point cloud. The red zoom-in detail highlights that our reconstruction captures the vehicle and human posture details most accurately and completely. In contrast, the Pointsoup method merges the human point cloud with the background, and other methods fail to fully reconstruct the human posture. In Figure 9, the red zoom-in clearly reveals the contours of the chair's backrest, which are not as distinct in the reconstructions from IPDAE and Pointsoup. The blue zoom-in detail further demonstrates that our method reconstructs the continuous chair legs, whereas other methods exhibit gaps or missing parts. In Figure 10, both the red and blue zoom-in details show that our method successfully reconstructs the chair's backrest, while the backrest in the IPDAE and Pointsoup reconstructions appears relatively blurry. Additionally, Figures 8–10 include bitrate and D1 PSNR comparisons, clearly demonstrating that our method achieves the best reconstruction quality and detail at the lowest bitrate across all tested scene point clouds.


(a) Ground Truth

(b) G-PCCv23(O) 0.769/60.370dB

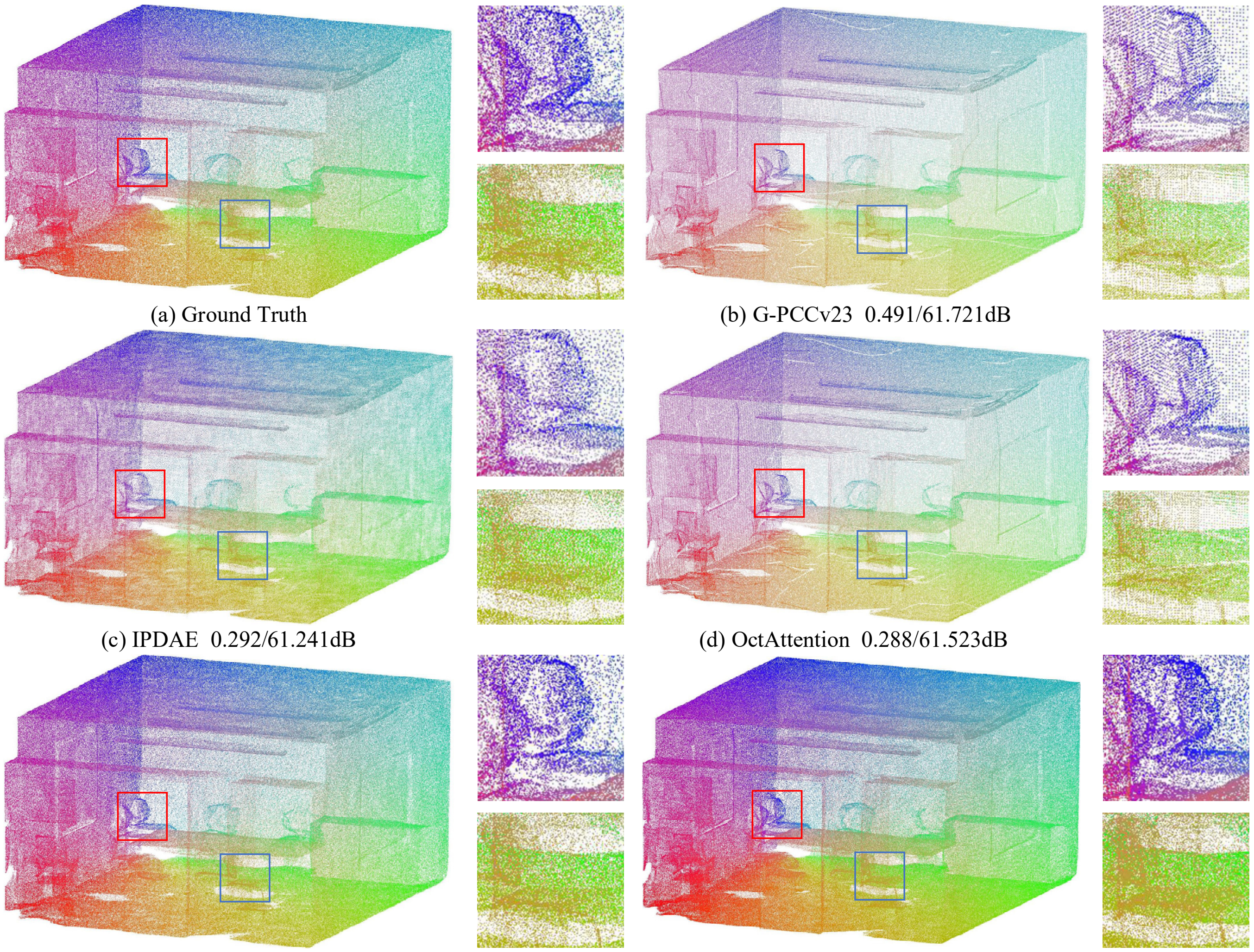
(c) IPDAE 0.641/56.937dB

(d) OctAttention 0.539/58.95dB

(e) Pointsoup 0.467/59.304dB

(f) Ours 0.45/60.016dB

Figure 8. Visual comparison of KITTI dataset. The red and blue boxes represent the magnified details of the decompressed point cloud.



(a) Ground Truth

(b) G-PCCv23 0.491/61.721dB

(c) IPDAE 0.292/61.241dB

(d) OctAttention 0.288/61.523dB

(e) Pointsoup 0.304/62.205dB

(f) Ours 0.275/62.215dB

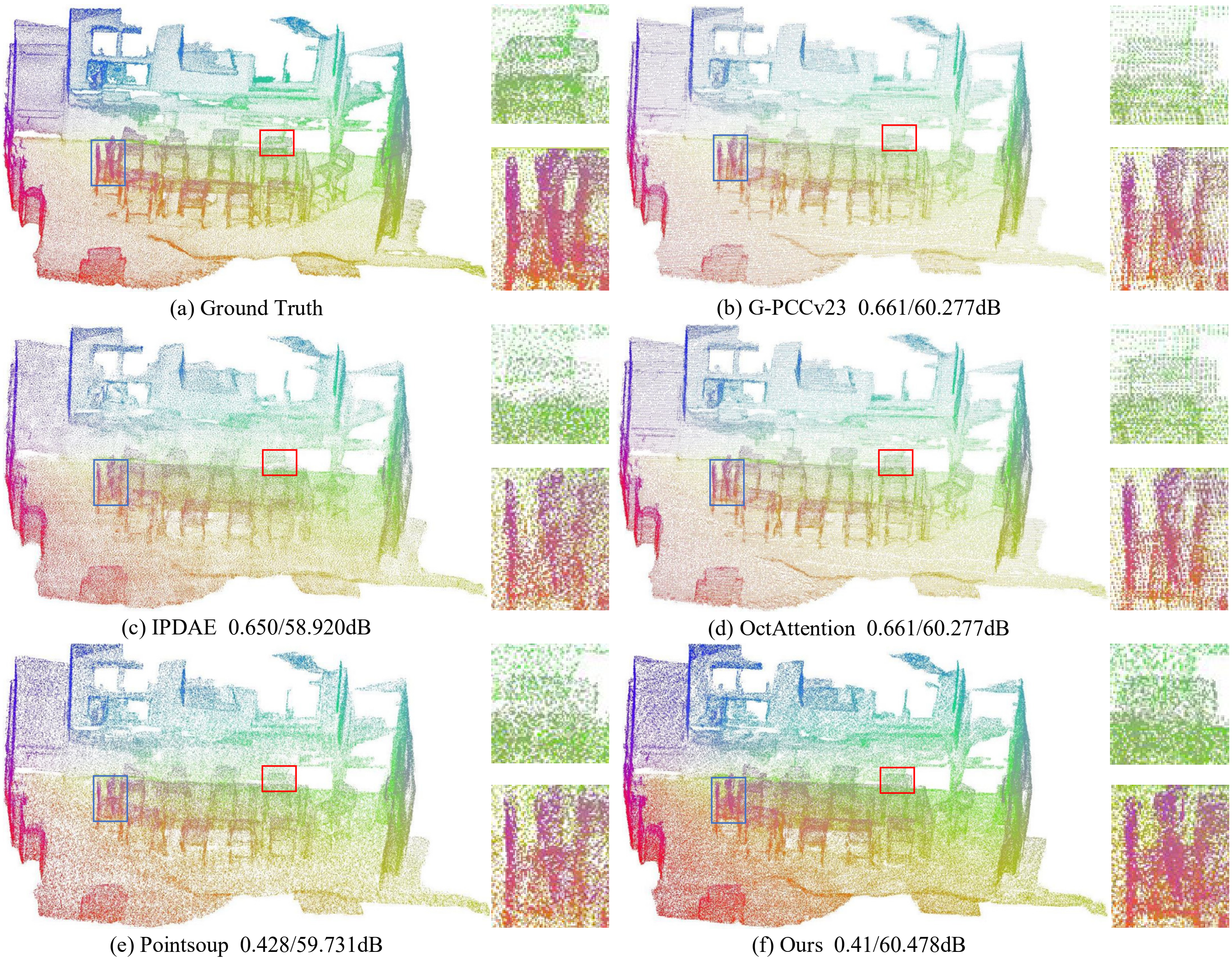
Figure 9. Visual comparison of S3DIS datasets. The red and blue boxes represent the magnified details of the decompressed point cloud.

4.3. Ablation Experiment

Comparative experiments on different datasets have demonstrated the effectiveness of our algorithm. To further validate the contributions of each module in the proposed network, we conducted ablation studies. First, we performed ablation on the cross-attention module to demonstrate its feature extraction capability. Next, we ablated the efficient generation module to verify its effectiveness in improving point cloud reconstruction quality. During testing, the window size was set to 128, and we compared the bpp and D1 PSNR results of different ablation models, as shown in Table 5. From Table 5, it is evident that both the proposed cross-attention and optimization generation modules are effective for sparse point cloud geometry compression.

Table 5. Module ablation comparison.

Cross-Attention Module	Efficient Generation Module	bpp	D1 PSNR
√	×	0.48	59.98 dB
×	√	0.45	59.77 dB
√	√	**0.45**	**60.02 dB**


(a) Ground Truth

(b) G-PCCv23 0.661/60.277dB

(c) IPDAE 0.650/58.920dB

(d) OctAttention 0.661/60.277dB

(e) Pointsoup 0.428/59.731dB

(f) Ours 0.41/60.478dB

Figure 10. Visual comparison of ScanNet datasets. The red and blue boxes represent the magnified details of the decompressed point cloud.

For a more intuitive presentation of the ablation study results, we visualized them in Figure 11. When only the proposed optimization generation module is used, it enhances reconstruction quality while reducing the bitrate, as indicated by the brown line in Figure 11. When only the positional attention module is applied, it significantly improves reconstruction quality with a slight increase in bitrate, as shown by the green line in Figure 11. When both modules are incorporated, the network achieves substantial improvements in reconstruction quality while also reducing bitrate, as represented by the red line in Figure 11. In summary, these results strongly demonstrate the effectiveness of each module in the proposed algorithm.

We also compared the encoding and decoding times for different ablation models, as shown in Table 6. The results indicate no significant changes in encoding or decoding times due to module modifications, suggesting that the improved compression performance of the proposed method does not come at the cost of increased computation or parameter size.

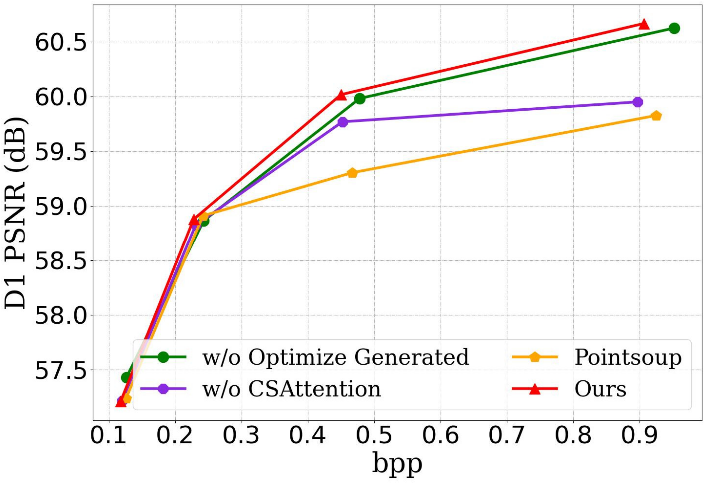


Figure 11. Comparison of module ablation bitrate–distortion curves.

Table 6. Codec time comparison.

	w/o Cross-Attention Module	**w/o Efficient Generation Module**	**Ours**
Codec Time/s	1.46/0.04	1.48/0.04	1.49/0.04

5. Discussion

The proposed large-scale point cloud geometry compression algorithm addresses the issue of long terminal decoding times by employing an efficient point cloud generation module. However, the encoding time for the method remains relatively long, and future research will focus on optimizing this aspect. Comparative analysis of point-based large-scale point cloud geometry compression algorithms indicates that, while the proposed algorithm outperforms the latest methods, the PSNR for bpp values within the 0–1 range does not exceed 63 dB across the three datasets. This suggests that there is substantial room for improvement in large-scale point cloud geometry compression. Future work will explore more efficient network structures to achieve sparse point cloud geometry compression with higher decoding accuracy while maintaining fast encoding and decoding speeds, promoting broader applications of large-scale point clouds.

6. Conclusions

The proposed large-scale scene point cloud geometry compression algorithm achieves state-of-the-art compression performance with the fastest decoding time. In the encoder stage, geometric features are extracted from local windows using a cross-attention module, and dilated windows are introduced as cross-scale priors to enable parallel inference of the quantized feature distribution. These features are then binary-encoded using an entropy coding module. During decoding, the geometric features are first refined, and an efficient optimized generation module is employed to reconstruct the point cloud coordinates. Finally, a reverse alignment operation restores the point cloud in each window to its original scale. Extensive comparative and ablation experiments demonstrate that our method outperforms benchmark algorithms across three open source datasets. Additionally, the proposed algorithm's neural model size is only 2.8 MB, providing valuable insights for the deployment of large-scale point cloud encoding and decoding technology on mobile devices.

Author Contributions: Conceptualization, S.L.; data curation, S.L.; formal analysis, S.L.; funding acquisition, C.H. and H.Y.; investigation, S.L.; methodology, S.L.; project administration, C.H. and H.Y.; resources, C.H. and H.Y.; writing—original draft, S.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been partially supported by the National Key R&D Program of China under grant No. 2020YFB1709200.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: The dataset used in this paper is the publicly available ShapeNat, S3DIS, ScanNet, and KITTI. They can be downloaded at the following links: https://shapenet.org/; https://redivis.com/datasets/9q3m-9w5pa1a2h; https://www.scan-net.org/; https://www.cvlibs.net/datasets/kitti/ (accessed on 31 October 2024).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Chen, S.; Liu, B.; Feng, C.; Vallespi-Gonzalez, C.; Wellington, C. 3d point cloud processing and learning for autonomous driving: Impacting map creation, localization, and perception. *IEEE Signal Process. Mag.* **2020**, *38*, 68–86. [CrossRef]
2. Christen, S.; Yang, W.; Pérez-D'Arpino, C.; Hilliges, O.; Fox, D.; Chao, Y.W. Learning human-to-robot handovers from point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 9654–9664.
3. Yu, S.; Sun, S.; Yan, W.; Liu, G.; Li, X. A method based on curvature and hierarchical strategy for dynamic point cloud compression in augmented and virtual reality system. *Sensors* **2022**, *22*, 1262. [CrossRef] [PubMed]
4. Graziosi, D.; Nakagami, O.; Kuma, S.; Zaghetto, A.; Suzuki, T.; Tabatabai, A. An overview of ongoing point cloud compression standardization activities: Video-based (V-PCC) and geometry-based (G-PCC). *APSIPA Trans. Signal Inf. Process.* **2020**, *9*, e13. [CrossRef]
5. Wang, J.; Zhu, H.; Liu, H.; Ma, Z. Lossy point cloud geometry compression via end-to-end learning. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *31*, 4909–4923. [CrossRef]
6. Wang, J.; Ding, D.; Li, Z.; Ma, Z. Multiscale Point Cloud Geometry Compression. In Proceedings of the Data Compression Conference, Snowbird, UT, USA, 23–26 March 2021; pp. 73–82.
7. You, K.; Gao, P.; Li, Q. IPDAE: Improved Patch-Based Deep Autoencoder for Lossy Point Cloud Geometry Compression. In Proceedings of the 1st International Workshop on Advances in Point Cloud Compression, Processing and Analysis, New York, NY, USA, 14 October 2022; pp. 1–10.
8. Lu, S.; Yang, H.; Han, C. TransPCGC: Point Cloud Geometry Compression Based on Transformers. *Algorithms* **2023**, *16*, 484. [CrossRef]
9. Nguyen, D.T.; Quach, M.; Valenzise, G.; Duhamel, P. Learning-based lossless compression of 3d point cloud geometry. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Toronto, ON, Canada, 6–11 June 2021; pp. 4220–4224.
10. Nguyen, D.T.; Quach, M.; Valenzise, G.; Duhamel, P. Multiscale deep context modeling for lossless point cloud geometry compression. In Proceedings of the IEEE International Conference on Multimedia & Expo Workshops, Shenzhen, China, 5–9 July 2021; pp. 1–6.
11. Nguyen, D.T.; Quach, M.; Valenzise, G.; Duhamel, P. Lossless coding of point cloud geometry using a deep generative model. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *31*, 4617–4629. [CrossRef]
12. Wang, J.; Ding, D.; Li, Z.; Feng, X.; Cao, C.; Ma, Z. Sparse tensor-based multiscale representation for point cloud geometry compression. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 9055–9071. [CrossRef] [PubMed]
13. Huang, L.; Wang, S.; Wong, K.; Liu, J.; Urtasun, R. Octsqueeze: Octree-structured entropy model for lidar compression. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 1313–1323.
14. Cui, M.; Long, J.; Feng, M.; Li, B.; Kai, H. OctFormer: Efficient octree-based transformer for point cloud compression with local enhancement. In Proceedings of the AAAI Conference on Artificial Intelligence, Washington, DC, USA, 7–14 February 2023; Volume 37, pp. 470–478.

15. Que, Z.; Lu, G.; Xu, D. Voxelcontext-net: An octree based framework for point cloud compression. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 6042–6051.
16. Fu, C.; Li, G.; Song, R.; Gao, W.; Liu, S. Octattention: Octree-based large-scale contexts model for point cloud compression. In Proceedings of the AAAI Conference on Artificial Intelligence, Online, 22 February–1 March 2022; Volume 36, pp. 625–633.
17. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660.
18. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1–10.
19. Huang, T.; Zhang, J.; Chen, J.; Ding, Z.; Tai, Y.; Zhang, Z.; Wang, C.; Liu, Y. 3QNet: 3D Point Cloud Geometry Quantization Compression Network. *ACM Trans. Graph.* **2022**, *41*, 1–13. [CrossRef]
20. Wiesmann, L.; Milioto, A.; Chen, X.; Stachniss, C.; Behley, J. Deep compression for dense point cloud maps. *IEEE Robot. Autom. Lett.* **2021**, *6*, 2060–2067. [CrossRef]
21. You, K.; Liu, K.; Yu, L.; Gao, P.; Ding, D. Pointsoup: High-Performance and Extremely Low-Decoding-Latency Learned Geometry Codec for Large-Scale Point Cloud Scenes. *arXiv* **2024**, arXiv:2404.13550.
22. Liang, Z.; Liang, F. TransPCC: Towards Deep Point Cloud Compression via Transformers. In Proceedings of the 2022 International Conference on Multimedia Retrieval, Newark, NJ, USA, 27–30 June 2022; pp. 1–5.
23. Pang, J.; Lodhi, M.A.; Tian, D. GRASP-Net: Geometric residual analysis and synthesis for point cloud compression. In Proceedings of the 1st International Workshop on Advances in Point Cloud Compression, Processing and Analysis, Lisboa, Portugal, 14 October 2022; pp. 11–19.
24. Pang, J.; Bui, K.; Tian, D. PIVOT-Net: Heterogeneous Point-Voxel-Tree-based Framework for Point Cloud Compression. In Proceedings of the International Conference on 3D Vision, Davos, Switzerland, 18–21 March 2024; pp. 1270–1279.
25. Sun, L.; Wang, J.; Shi, Y.; Zhu, Q.; Yin, B.; Ling, N. Octree-Based Temporal-Spatial Context Entropy Model for LiDAR Point Cloud Compression. In Proceedings of the 2023 IEEE International Conference on Visual Communications and Image Processing (VCIP), Jeju, Republic of Korea, 4–7 December 2023; pp. 1–5.
26. Fan, T.; Gao, L.; Xu, Y.; Wang, D.; Li, Z. Multiscale latent-guided entropy model for lidar point cloud compression. *IEEE Trans. Circuits Syst. Video Technol.* **2023**, *33*, 7857–7869. [CrossRef]
27. Huang, R.; Wang, M. Patch-Wise LiDAR Point Cloud Geometry Compression Based on Autoencoder. In *Image and Graphics*; Springer Nature Switzerland: Cham, Switzerland, 2023; pp. 299–310.
28. Wang, M.; Huang, R.; Dong, H.; Lin, D.; Song, Y.; Xie, W. msLPCC: A Multimodal-Driven Scalable Framework for Deep LiDAR Point Cloud Compression. In Proceedings of the AAAI Conference on Artificial Intelligence, Washington, DC, USA, 20–27 February 2024; Volume 38, pp. 5526–5534.
29. Biswas, S.; Liu, J.; Wong, K.; Wang, S.; Urtasun, R. Muscle: Multi sweep compression of lidar using deep entropy models. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 22170–22181.
30. Song, R.; Fu, C.; Liu, S.; Li, G. Efficient hierarchical entropy model for learned point cloud compression. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 14368–14377.
31. Song, R.; Fu, C.; Liu, S.; Li, G. Large-Scale Spatio-Temporal Attention Based Entropy Model for Point Cloud Compression. In Proceedings of the IEEE International Conference on Multimedia and Expo, Brisbane, Australia, 10–14 July 2023; pp. 2003–2008.
32. Lodhi, M.A.; Pang, J.; Tian, D. Sparse convolution based octree feature propagation for lidar point cloud compression. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Rhodes Island, Greece, 4–10 June 2023; pp. 1–5.
33. Luo, A.; Song, L.; Nonaka, K.; Unno, K.; Sun, H.; Goto, M.; Katto, J. SCP: Spherical-Coordinate-Based Learned Point Cloud Compression. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 20–27 February 2024; Volume 38, pp. 3954–3962.
34. Lang, I.; Manor, A.; Avidan, S. Samplenet: Differentiable point cloud sampling. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 7578–7588.
35. Hu, Q.; Yang, B.; Xie, L.; Rosa, S.; Guo, Y.; Wang, Z.; Trigoni, N.; Markham, A. Learning semantic segmentation of large-scale point clouds with random sampling. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 8338–8354. [CrossRef]
36. Connor, M.; Kumar, P. Fast construction of k-nearest neighbor graphs for point clouds. *IEEE Trans. Vis. Comput. Graph.* **2010**, *16*, 599–608. [CrossRef] [PubMed]
37. Wu, T.; Pan, L.; Zhang, J.; Wang, T.; Liu, Z.; Lin, D. Density-aware chamfer distance as a comprehensive metric for point cloud completion. In Proceedings of the 35th International Conference on Neural Information Processing Systems, Red Hook, NY, USA, 6–14 December 2021; pp. 29088–29100.
38. Chang, A.X.; Funkhouser, T.; Guibas, L.; Hanrahan, P.; Huang, Q.; Li, Z.; Savarese, S.; Savva, M.; Song, S.; Su, H.; et al. Shapenet: An information-rich 3D model repository. *arXiv* **2015**, arXiv:1512.03012.

39. Armeni, I.; Sener, O.; Zamir, A.R.; Jiang, H.; Brilakis, I.; Fischer, M.; Savarese, S. 3D semantic parsing of large-scale indoor spaces. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1534–1543.
40. Dai, A.; Chang, A.X.; Savva, M.; Halber, M.; Funkhouser, T.; Nießner, M. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 5828–5839.
41. Behley, J.; Garbade, M.; Milioto, A.; Quenzel, J.; Behnke, S.; Stachniss, C.; Gall, J. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 9297–9307.
42. Kingma, D.P. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

MDPI AG
Grosspeteranlage 5
4052 Basel
Switzerland
Tel.: +41 61 683 77 34

Sensors Editorial Office
E-mail: sensors@mdpi.com
www.mdpi.com/journal/sensors

Disclaimer/Publisher's Note: The title and front matter of this reprint are at the discretion of the Guest Editors. The publisher is not responsible for their content or any associated concerns. The statements, opinions and data contained in all individual articles are solely those of the individual Editors and contributors and not of MDPI. MDPI disclaims responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

MDPI

Academic Open
Access Publishing

mdpi.com

ISBN 978-3-7258-4688-7