GNNs-Based 3D Object Detection in Autonomous Driving: System and Empirical Evaluation

1st Thai Anh Vo
Phenikaa School of Computing
Phenikaa University
Hanoi, Vietnam
anh.vothai@phenikaa-uni.edu.vn

2nd Lan Anh Nguyen
Computer Science and Engineering
Chung-Ang University
Seoul, Korea
loglamo@cau.ac.kr

3rd Trung Son Doan
Phenikaa School of Computing
Phenikaa University
Hanoi, Vietnam
son.doantrung@phenikaa-uni.edu.vn

4th Son Hong Ngo
Phenikaa School of Computing
Phenikaa University
Hanoi, Vietnam
son.ngohong@phenikaa-uni.edu.vn

***Abstract*—Accurate 3D object detection from point clouds is essential for autonomous driving, yet remains difficult due to the sparsity and irregularity of Light Detection and Ranging (LiDAR) data. In this work, we introduce a 3D object detection framework based on Graph Neural Networks (GNNs) designed for autonomous driving scenarios. To assess its effectiveness, we implement PGD, a representative GNN-based model, and compare it with three widely adopted alternatives, such as PointPillar, CenterPoint, and SECOND. We conduct one of the first cross-dataset empirical studies of GNN-based detection across the KITTI, nuScenes, and Waymo benchmarks. Our results show that the GNN-based approach (e.g., PGD) underperforms in terms of accuracy and exhibits inefficiencies in training time and memory usage. These findings highlight the potential of GNNs and point to promising directions for future improvements in 3D object detection.**

***Index Terms*—3D object detection, autonomous driving, point clouds, neural networks, GNNs, PGD, PointNet.**

I. Introduction

Autonomous driving plays a pivotal role in transforming transportation and improving safety, efficiency, and accessibility for individuals and society as a whole [1]. As the technology continues to evolve, it has garnered significant attention from both the industry and the research community, driven by its potential to revolutionize mobility. A fundamental aspect of autonomous driving is object detection, which enables vehicles to perceive and interpret their environment in real time. Through the detection of objects such as pedestrians, vehicles, traffic signs, and various obstacles, object detection systems are essential for ensuring safe navigation and effective decision-making. Traditionally, 2D images have been widely used in the object detection component of autonomous driving systems.

However, conventional 2D imaging fails to capture depth and structural information [2]–[4], which makes achieving accurate and reliable object detection difficult. In contrast, 3D point cloud images, generated by systems such as Light Detection and Ranging (LiDAR), Structured Light Sensors, and Time-of-Flight (ToF) Cameras, overcome these limitations by enhancing depth perception, spatial awareness, and overall robustness [5]–[7].

However, the inherent characteristics of point cloud images, such as the irregularity of 3D data, data sparsity, and unscalability, present challenges in the model learning process (e.g., Convolutional Neural Networks (CNNs)) for object detection [8]–[11]. Graph Neural Networks (GNNs) offer an effective solution by utilizing graph structures to address these issues [8], [12], [13] in theory. GNNs are particularly adept at capturing relationships in unstructured data and provide improved scalability and performance in processing 3D point clouds.

This study examines the current landscape and future potential of GNN-based 3D object detection for autonomous driving. We begin by designing a system architecture tailored to GNN-based 3D detection in this domain. We then implement and evaluate its performance using multiple benchmark datasets, comparing it against established deep learning-based methods such as PointPillar, CenterPoint, and SECOND, with PGD representing the GNN-based approach.

Section II outlines the system design for GNN-based 3D object detection in autonomous driving, along with the datasets and detection methods employed for empirical evaluation. The implementation details and experimental analysis are presented in Section III.

II. Methodology

We provide an overview of GNNs based 3D object detection system in II-A, GNNs based 3D object detection in autonomous driving in II-B, datasets, and 3D detection methods used for empirical evaluation in II-D.

A. Primary components in GNNs-based 3D object detection

The figure 1 propvides primary components of 3D object detection methods utilizing GNNs. These methods take point cloud data as input, which can be structured as grids, sets,

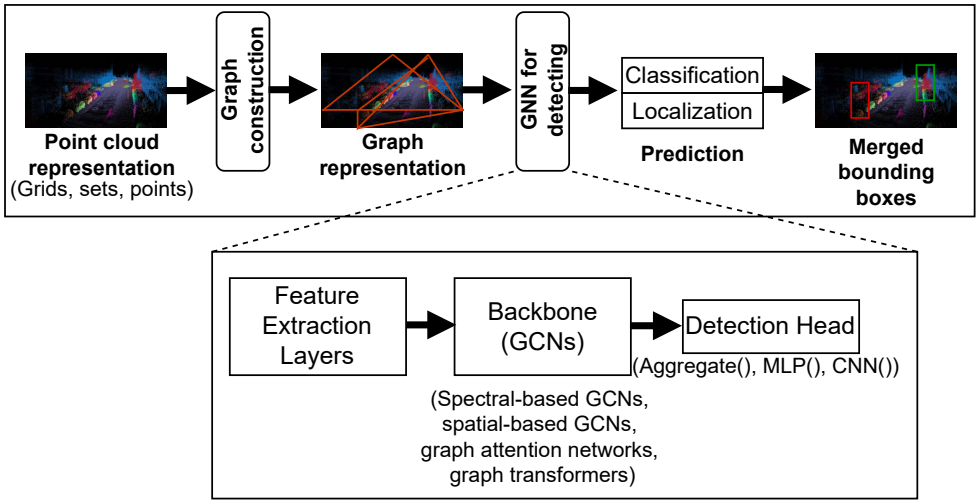

Fig. 1. Primary components in GNNs-based 3D object detection based on GNNs

or individual points. The **'Graph construction'** component processes this point cloud representation to generate graphs. Subsequently, the **'GNN for detecting'** component applies various graph-based techniques, such as spectral-based GCNs, spatial-based GCNs, graph attention networks, and graph transformers, to analyze the graph representation and predict object classification and localization. Finally, the detected objects are displayed within merged bounding boxes.

B. GNNs-based 3D object detection in autonomous driving

Figure 2 illustrates a system for 3D object detection in autonomous driving, utilizing GNNs.
Vehicles used in autonomous driving, such as cars and trucks, are equipped with devices like LiDAR that generate point cloud data. A 3D object detection service processes this point cloud data as input and applies a 3D object detector to detect objects, producing outputs with bounding boxes. The detected 3D objects are then forwarded to a decision-making unit for further processing in autonomous driving tasks. The 3D object detector consists of primary components as explained in **'Primary components in GNNs-based 3D object detection'** in II-B.

C. Datasets for autonomous driving

In this section, we provide information of datasets containing 3D objects relevant to autonomous driving, including pedestrians, various types of vehicles (such as cars, trucks, buses, motorcycles, and bicycles), and traffic-related elements (such as traffic signs, signals, road barriers, cones, and lane markings). Table I presents the key characteristics of the three datasets used in our experiments.

The table I presents the number of scenes, object classes, and bounding boxes, along with the scene types and sensor types used. All datasets feature urban driving scenarios. Moreover, the images are captured using RGB cameras and LiDAR, with the latter providing data in point cloud format. Consequently, these datasets are well-suited for 3D object detection in autonomous driving.

D. 3D object detection methods for comparison

Due to the variety of 3D object detection methods, this work focuses on several widely used methods for comparison, including PointPillar, CenterPoint, SECOND, and PGD. Table II provides key informations (e.g., region proposal, single shot, view type, representation, GNNs) of these methods. Key features of these methods are:

- **PointPillar** [17]: The core of PointPillar is the Pillar Feature Encoding (PFE) module. It converts the point cloud within each pillar into a fixed-size feature vector. This step involves aggregating point-level features (such as intensity, height, and coordinates) into a structured format that can be processed by deep learning networks. For 2D convolutional backbone, after the point cloud is transformed into a pillar-based grid, the data is fed into a 2D convolutional neural network (CNN). The CNN processes the bird's-eye view (BEV) of the point cloud, extracting spatial features and learning to detect objects in the scene. For object Detection, the processed features are passed to a detection head that predicts the 3D bounding boxes of objects (such as cars, pedestrians, cyclists) and their associated attributes (e.g., orientation, dimensions). PointPillar directly outputs the detection results in a single-shot process, making it fast and suitable for real-time applications.
- **CenterPoint** [18]: The core idea behind CenterPoint is to predict the center of each 3D object. The method treats object detection as a center-based localization problem, where the model directly regresses the object's center position, size, orientation, and other attributes without requiring region proposals. Besides, the key innovation

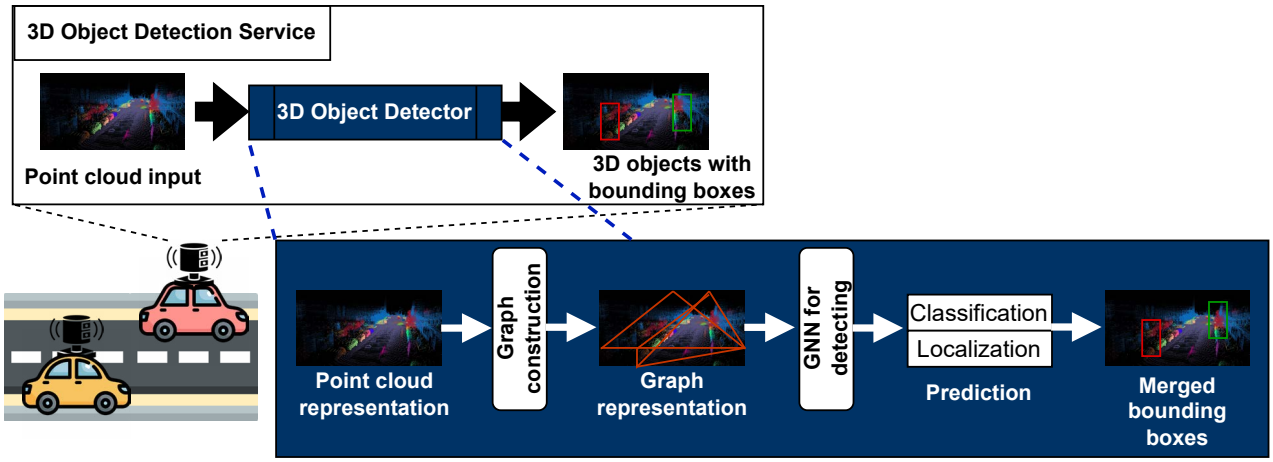

Fig. 2. GNNs based 3D object detection in autonomous driving

TABLE I
3D DATASETS FOR AUTONOMOUS DRIVING (#: INDICATES THE NUMBER OF SOMETHING)

	Year	#Scences	#Classes	#3D Boxes	#Scence type	#Sensors
KITTY [14]	2012	22	8	200K	Urban driving	RGB & LiDAR
nuScences [15]	2020	1K	23	1.4M	Urban driving	RGB & LiDAR
Waymo [16]	2020	1150	Over 12	12M	Roadway	RGB & LiDAR

TABLE II
3D OBJECT DETECTION METHODS FOR COMPARISON ('REGION PROPOSAL' INDICATES WHETHER THE METHOD IS BASED ON REGION PROPOSALS FOR 3D OBJECT DETECTION; 'SINGLE SHOT' SPECIFIES WHETHER THE METHOD IS A SINGLE-SHOT 3D OBJECT DETECTION APPROACH; 'VIEW TYPE' DESCRIBES THE TYPE OF VIEW USED AS INPUT TO THE METHOD (E.G., BIRD'S-EYE VIEW (BEV) OR RAW POINTS (POINT)); 'REPRESENTATION' SHOWS WHETHER THE INPUT IS PRESENTED AS A GRID, SET, OR POINT FOR FURTHER PROCESSING; 'GNNS' INDICATES WHETHER THE METHOD UTILIZES GRAPH NEURAL NETWORKS (GNNS) FOR DETECTION AND SPECIFIES WHICH GNN IS USED.)

	Region proposal	Single shot	View type	Representation	GNNs
PointPillar [17]		✓	BEV	Grid	
CenterPoint [18]		✓	BEV	Grid	
SECOND [19]	✓		BEV	Grid	
PGD [20]	✓		Point	Point	✓, PointNet++

in CenterPoint is the center heatmap. For each object, the model generates a heatmap where the center of the object is highlighted. This allows the network to focus on the object center as the key reference point for bounding box prediction. For 2D convolutional backbone, after converting the point cloud into a BEV grid, CenterPoint uses a 2D convolutional neural network (CNN) to extract features from the BEV representation. These features are then used to predict the object centers and other attributes (like orientation, size, and velocity) in a single-shot manner. For single-Shot detection, similar to PointPillar, CenterPoint is a single-shot detection method. It directly predicts 3D bounding boxes and object attributes in one pass, making it computationally efficient and fast.

- **SECOND** [19]: SECOND begins by converting the raw LiDAR point cloud into a voxel grid representation. The point cloud is divided into small 3D voxels, where each voxel represents a small region in 3D space. This is done to process the point cloud in a structured manner, making it suitable for convolutional operations. To handle the sparsity of the voxelized data, SECOND uses sparse 3D convolutions instead of regular dense convolutions. Sparse convolutions focus only on non-empty voxels, significantly reducing computational cost while maintaining high accuracy in feature extraction. SECOND employs a region proposal network (RPN) in its first stage. After voxelizing the point cloud, the model generates candidate regions (proposals) where potential objects might be located. These proposals are based on the sparse 3D features extracted from the voxel grid. For object detection, in the second stage, the proposed regions are refined using fully connected layers to predict the 3D bounding boxes and object attributes (such as class, orientation, and size).
- **PGD** [20]: PGD works directly on the raw LiDAR point cloud and constructs a graph where each point in the cloud is a node, and the edges between nodes represent spatial relationships or geometric dependencies between

neighboring points. Similar to other 3D object detection methods, PGD generates region proposals that represent potential object locations. The GNN learns to detect object centers and regions by examining the graph structure and the relationships between points. Like PointPillar and CenterPoint, PGD is a single-shot detection method, meaning it predicts the 3D bounding boxes and object attributes (such as size, orientation, and class) in a single forward pass, making it efficient and suitable for real-time applications.

In this section, we present the system design for GNN-based 3D object detection in autonomous driving. Additionally, we outline the datasets and 3D object detection methods selected for comparison and analysis. In the following section, we conduct experiments and provide an in-depth analysis of the datasets and detection methods.

III. Experiments and Results

A. Experimental Setups

The methods PointPillar, CenterPoint, SECOND, and PGD, discussed in the previous section, are implemented in our experiments for comparison. We conduct experiments using the three datasets: KITTI, nuScenes, and Waymo. We perform training on NVIDIA GTX 1080Ti GPUs with 9.2GB of memory.

B. Results and Analysis

1) Performance of 3D object detection methods on KITTI dataset: The KITTI dataset serves as a fundamental benchmark for evaluating 3D object detection models, particularly under moderate-difficulty conditions.

Table 3 shows model performance across three critical object categories of KITTI: cars, pedestrians, and cyclists, using the standard metrics of Car@R11, Pedestrian@R11, and Cyclist@R11. This metric evaluates the model's ability to detect cars. The '@R11' likely refers to the recall level at which precision is measured, typically using an 11-point interpolation method for calculating Average Precision (AP). The value represents the AP score for car detection under moderate difficulty conditions.

The results indicate that SECOND achieves superior performance across all categories, establishing it as a strong baseline for comparison. PointPillar demonstrates competitive car detection capabilities but exhibits performance degradation in pedestrian and cyclist detection scenarios. CenterPoint maintains moderate performance across all categories, while the GNNs-based method, **PGD, though innovative in its GNN-based approach, currently achieves lower detection accuracy than established methods**.

2) Performance of 3D object detection on Waymo dataset: The Waymo Dataset [16] presents a more complex evaluation environment, featuring multiple object categories and distinct difficulty levels (e.g., L1 and L2). L1 means Level 1, indicating objects with more than 5 LiDAR points in their bounding box. L2 means Level 2, indicating objects with at least 1 LiDAR point in their bounding box. The levels, Vec_L1, Vec_L2, Ped_L1, Ped_L2, Cyc_L1, and Cyc_L2, showed in Table 4, capture model difficulty levels across classes.

The difficulty levels effectively distinguish between detection challenges, enabling a nuanced understanding of how each model performs under different conditions. The separation of object categories allows for targeted performance analysis, helping researchers identify specific strengths and weaknesses in various detection scenarios. The metrics align with Waymo's established evaluation standards, ensuring comparability with other work using this benchmark.

CenterPoint demonstrates strong performance across most metrics, particularly for vehicle detection at both difficulty levels, while PointPillar shows reliable performance in vehicle detection but variability in pedestrian and cyclist scenarios. SECOND maintains consistent performance across all categories, while **PGD continues to show the lowest detection accuracy among compared models, though with potential for improvement through architectural refinement**.

3) Performance of 3D object detection on nuScences dataset: The nuScenes dataset provides a comprehensive evaluation framework with its unique set of metrics including mATE (mean Average Translation Error, the average Euclidean distance between the predicted and ground-truth object centers in 3D space), mASE (mean Average Scale Error, this evaluates the accuracy of the predicted object dimensions), mAOE (mean Average Orientation Error, this assesses the accuracy of the predicted object orientation), mAVE (mean Average Velocity Error, this evaluates the accuracy of the predicted object velocities), mAAE (mean Average Attribute Error, this assesses the accuracy of attribute predictions) These metrics assess various aspects of detection quality, including localization accuracy, scale estimation, orientation accuracy, velocity estimation, and attribute classification. The results in Table 5 demonstrate how different models perform across these metrics.

PointPillar shows reasonable performance across most metrics but lags in orientation and attribute accuracy. CenterPoint demonstrates improved performance in translation and scale errors but shows variability in orientation and attribute metrics. SECOND achieves a good balance across most metrics, while **PGD exhibits the lowest performance among the compared models**.

PGD's performance limitations on nuScenes can be attributed to several factors. The multi-modal nature of nuScenes requires sophisticated graph attention mechanisms to effectively model the diverse interactions between vehicles, pedestrians, and cyclists in urban environments. PGD's current architecture might not yet fully capture these complex relationships, particularly for attribute and velocity predictions. The model may also struggle with the high degree of occlusion present in nuScenes scenes, where effective graph message passing is crucial for inferring occluded object properties. Additionally, the current implementation might not optimally balance the trade-off between localization accuracy and computational efficiency, affecting its performance on metrics like mATE and mASE. The nuScenes evaluation protocol

TABLE III
ACCURACY ACROSS METHODS ON KITTI DATASET

Model	**Car@R11(%)**	**Pedestrial@R11(%)**	**Cyclist@R11(%)**
PointPillar	38.77	29.58	48.60
CenterPoint	59.19	55.74	51.32
SECOND	**68.92**	**53.98**	**67.15**
PGD	18.61	18.90	19.47

TABLE IV
ACCURACY ACROSS METHODS ON WAYMO DATASET

Model	**Vec_L1(%)**	**Vec_L2(%)**	**Ped_L1(%)**	**Ped_L2(%)**	**Cyc_L1(%)**	**Cyc_L2(%)**
PointPillar	70.21/70.32	65.35/62.52	55.32/45.23	55.12/54.90	63.12/55.34	60.24/58.23
CenterPoint	**71.90/72.43**	**66.21/63.16**	**72.92/66.28**	**64.32/55.90**	**65.50/54.30**	**61.40/57.35**
SECOND	63.68/57.54	64.30/62.21	60.49/52.09	58.20/52.32	59.23/55.21	59.32/56.32
PGD	39.32/30.12	32.20/30.32	29.30/27.32	30.20/28.90	31.25/28.20	28.30/26.20

TABLE V
ACCURACY ACROSS METHODS ON NUSCENCE DATASET

Model	**mATE(%)**	**mASE(%)**	**mAOE(%)**	**mAVE(%)**	**mAAE(%)**
PointPillar	39.32	36.52	52.21	39.17	40.60
CenterPoint	**54.14**	**57.84**	**52.92**	**52.20**	**56.10**
SECOND	50.87	52.20	53.23	49.90	52.32
PGD	32.23	31.54	30.20	31.25	32.10

appropriately tests model capabilities in complex urban driving scenarios. The multi-metric approach provides a well-rounded assessment of detection quality, ensuring models perform well across all aspects of 3D object detection.

4) Comprehensive comparison: The comprehensive comparison across all datasets, shown in Table III-B4, provides valuable insights into the relative performance of different models by incorporating multiple metrics mAP (mean Average Precision: This measures the overall precision of object detection across different confidence threshold), NDS (NuScenes Detection Score: This is a comprehensive score that combines the above metrics into a single value, with mAP weighted more heavily) alongside practical considerations such as training time and memory requirements. The combination of performance metrics and practical considerations offers a complete picture of each model's capabilities and implementation requirements.

In terms of accuracy, as measured by mAP and NDS, CenterPoint consistently outperforms the other methods. Contrary to expectations, the GNN-based method PGD yields the lowest performance across both metrics. This underperformance can be attributed to the inherent difficulty of constructing a meaningful graph directly from raw, unstructured point cloud data. Unlike PGD, rival methods such as PointPillar, CenterPoint, and SECOND first preprocess the input into a structured, grid-based representation (e.g., pillars or voxels). This critical step enables them to leverage the formidable feature extraction power of highly optimized Convolutional Neural Networks (CNNs). PGD's performance, in contrast, is fundamentally contingent upon the topological quality of its graph. If the constructed edges fail to accurately encode the essential geometric and contextual relationships between points, the efficacy of the GNN's message-passing mechanism is severely compromised, resulting in diminished detection accuracy, particularly for small, distant, or partially occluded objects. These results suggest limitations in the current model design of GNN-based 3D object detection approaches for autonomous driving, highlighting the need for future research to enhance their accuracy.

In terms of training time and memory consumption, PointPillar is the most efficient, requiring only 2.3 hours and 8.4 GB of memory. In contrast, CenterPoint and PGD exhibit the longest training durations. Notably, PGD also consumes the most memory on average. This is likely due to the intrinsic overhead of a graph-based paradigm. Unlike voxel-based methods that leverage efficient sparse convolutions on a reduced set of points, PGD must first perform a costly neighborhood search to construct a graph from tens of thousands of raw points. Subsequently, the GNN's message-passing operations across this large, densely connected graph demand significantly more computational resources and memory. This is compounded by the substantial storage required for the graph's adjacency matrix and per-node feature representations during training, making it far less efficient than its grid-based counterparts. These results indicate that PGD imposes higher computational and memory demands compared to other methods, underscoring its resource-related challenges. It needs to be address strongly in future.

IV. CONCLUSION

This study presents a GNN-based 3D object detection system tailored for autonomous driving. We conduct empirical evaluations across multiple datasets and detection methods to assess their accuracy and effectiveness. Our results show that the GNN-based approach (e.g., PGD) underperforms in terms of accuracy and exhibits inefficiencies in training time

TABLE VI
COMPREHENSIVE COMPARISON OF 3D OBJECT DETECTION MODELS

Model	Dataset	mAP(%)	NDS(%)	Training time (hours)	Training memory (GB)
PointPillar	**KITTI**	**70.80**	-	1.2	5.5
	NuScences	39.26	**53.26**	**2.3**	**8.4**
	Waymo	63.20	-	1.1	8.2
CenterPoint	KITTI	55.60	-	-	8.5
	NuScences	56.90	**65.27**	**3.2**	8.7
	Waymo	**65.40**	**-**	3.0	**9.2**
SECOND	**KITTI**	**78.30**	**-**	1.8	5.4
	NuScences	50.59	**53.20**	**3.2**	**8.7**
	Waymo	62.23	-	1.8	5.4
PGD	KITTI	18.33	-	2.2	9.1
	nuScences	**31.80**	42.50	**3.1**	**9.2**
	Waymo	29.32	-	2.2	**9.2**

and memory usage. Nonetheless, these findings highlight the potential of GNNs and point to promising directions for future improvements in 3D object detection.

REFERENCES

[1] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," *IEEE access*, vol. 8, pp. 58 443–58 469, 2020.

[2] T. Shen, Y. Xie, T. Yuan, and X. Zhang, "A detection methods with image recognition for specific obstacles in the urban rail area," *IEEE Access*, vol. 12, pp. 142 772–142 783, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:272897477

[3] D. Ristić-Durrant, M. Franke, and K. Michels, "A review of vision-based on-board obstacle detection and distance estimation in railways," *Sensors (Basel, Switzerland)*, vol. 21, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:235229629

[4] S. Syntakas, K. Vlachos, and A. Likas, "Object detection and navigation of a mobile robot by fusing laser and camera information," in *2022 30th Mediterranean Conference on Control and Automation (MED)*, 2022, pp. 557–563.

[5] M. H. Annaby, M. Mahmoud, H. A. Abdusalam, H. A. Ayad, and M. A. Rushdi, "2d representations of 3d point clouds via the stereographic projection with encryption applications," *Multim. Syst.*, vol. 30, p. 173, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:270412872

[6] W. Gao and G. Li, "Deep learning for 3d point clouds," 2025. [Online]. Available: https://api.semanticscholar.org/CorpusID:274775841

[7] Y. Lyu, X. Huang, and Z. Zhang, "Learning to segment 3d point clouds in 2d image space," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 12 255–12 264.

[8] D. Li, C. Lu, Z. Chen, J. Guan, J. Zhao, and J. Du, "Graph neural networks in point clouds: A survey," *Remote Sensing*, vol. 16, no. 14, p. 2518, 2024.

[9] K. A. Tychola, E. Vrochidou, and G. A. Papakostas, "Deep learning based computer vision under the prism of 3d point clouds: a systematic review," *The Visual Computer*, vol. 40, no. 11, pp. 8287–8329, 2024.

[10] H. Yan, A. Lau, and H. Fan, "Evaluating deep learning advances for point cloud semantic segmentation in urban environments," *KN-Journal of Cartography and Geographic Information*, pp. 1–20, 2025.

[11] Q. Cai, Y. Pan, T. Yao, and T. Mei, "3d cascade rcnn: High quality object detection in point clouds," *arXiv preprint arXiv:2211.08248*, 2022.

[12] W. Shi and R. Rajkumar, "Point-gnn: Graph neural network for 3d object detection in a point cloud," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 1711–1719.

[13] H. Zhou, W. Wang, G. Liu, and Q. Zhou, "Pointgat: Graph attention networks for 3d object detection," *Intelligent and Converged Networks*, vol. 3, no. 2, pp. 204–216, 2022.

[14] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[15] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.

[16] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov, "Scalability in perception for autonomous driving: Waymo open dataset," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 2443–2451.

[17] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 12 697–12 705.

[18] T. Yin, X. Zhou, and P. Krahenbuhl, "Center-based 3d object detection and tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 11 784–11 793.

[19] Y. Yan, Y. Mao, and B. Li, "Second: Sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, p. 3337, 2018.

[20] T. Wang, Z. Xinge, J. Pang, and D. Lin, "Probabilistic and geometric depth: Detecting objects in perspective," in *Conference on Robot Learning*. PMLR, 2022, pp. 1475–1485.