2025 IEEE/SICE International Symposium on System Integration (SII)
January 21-24, 2025, Munich, Germany

Long-Range Vehicle Detection of LiDAR using Adaboost Classification*

Na-Young Lim[1], Jeon-Hyeok Lee[2], Tae-Hyoung Park[3]

***Abstract*— In recent developments, autonomous racing has garnered attention as it aims to overcome the limitations of standard autonomous driving systems. Achieving safe racing conditions necessitates both fast and long-range perception. However, current 3D LiDAR object detection methods face challenges with high computational costs and limited detection ranges. These issues make them unsuitable for racing scenarios. To address these challenges, this paper proposes a clustering-based long-range vehicle detection method that relies solely on LiDAR. First, the method removes ground points and foreground points and clusters the remaining points. Subsequently, these clusters are classified as vehicles using AdaBoost, generating 2D bounding boxes in the range view. Experimental results demonstrate superior performance, achieving a computational efficiency of 53 Hz and a long-range detection accuracy of over 80%, compared to voxel-based and range-based methods. This approach offers a viable solution for autonomous racing environments.**

I. INTRODUCTION

Recently, autonomous racing has garnered significant attention as it pushes beyond the limits of conventional autonomous driving. For safe autonomous racing, rapid and extensive perception is essential. However, most existing methods focus solely on improving detection performance rather than extending detection range, making them unsuitable for autonomous racing.

LiDAR data inherently becomes sparser with increasing distance, as shown in Fig. 1, where the data for the same object varies based on distance. Consequently, traditional methods that focus on short-range detection struggle with long-range objects. Additionally, some methods fail to consider the trade-off between performance and computational time, making fast object detection challenging. These issues present substantial difficulties in directly applying conventional object detection methods to racing environments.

To address these challenges, methods have been proposed to enhance long-range object detection performance by utilizing additional sensors [1], [2]. Khoche et al. [1] utilizes a camera to supplement data for distant objects, while Wang et al. [2] enhances performance by complementing LiDAR data with radar and filtering objects through a camera. However, these approaches face issues with synchronization for multi-modal sensors and increased computational demands due to additional data. To address this issue, methods aimed at improving long-range object detection performance using only LiDAR sensors have been proposed. Zhang et al. [3] proposed a method that enhances detection performance by converting sparse features into dense features; however, this improvement comes at the cost of increased computational time. Conversely, Chen et al. [4] addresses the computational overhead by directly utilizing sparse features for efficient detection improvement. Nevertheless, it still fails to achieve significant performance gains for highly sparse long-range objects.

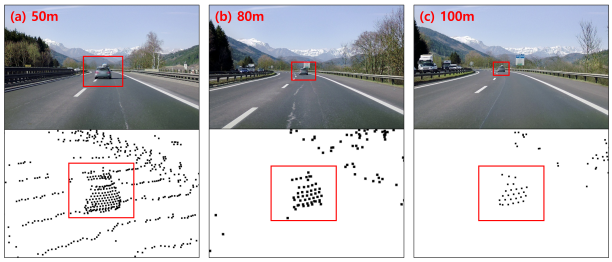

Fig. 1. **LiDAR and Camera data for distance** This is Camera and LiDAR data when the vehicle is located at 50m(a), is at 80m(b), is at 100m(c). As the distance increases, the LiDAR data becomes sparse.

Therefore, enhancing detection performance in racing environments remains a critical challenge, particularly in terms of reducing computational overhead and extending detection range. This paper further introduces a novel consideration specific to racing scenarios: accurate pose information for objects located more than 100 meters away is not essential. For short-range objects, precise pose information is crucial for predicting trajectories and avoiding collisions. In contrast, for long-range objects, preemptive maneuvers, such as lane changes, can prevent collisions, making detailed pose information less critical. The proposed method in this paper focuses on minimizing computational requirements while classifying and detecting objects at long ranges by estimating only their positions.

This paper presents a clustering-based long-range object detection method using only LiDAR data. Initially, preprocessing stage removes ground data and foreground data, followed by clustering to create clusters from the remaining data. Features are then extracted from these clusters, and AdaBoost is used to classify vehicle objects. Finally, range view-based bounding boxes are generated for these vehicles. As detailed pose information is not required, the final object

*This work was not supported by any organization.

[1]Na-Young Lim is with the Department of Intelligent Systems & Robotics, Chungbuk National University, Cheongju, South Korea. nayoung8218@chungbuk.ac.kr.

[2]Jeon-Hyeok Lee is with the Department of Control & Robot Engineering , Chungbuk National University, Cheongju, South Korea. jeon3359@naver.com.

[3]Tae-Hyoung Park is with the Department of Intelligent Systems & Robotics, Chungbuk National University, Cheongju, South Korea. taehpark@cbnu.ac.kr.

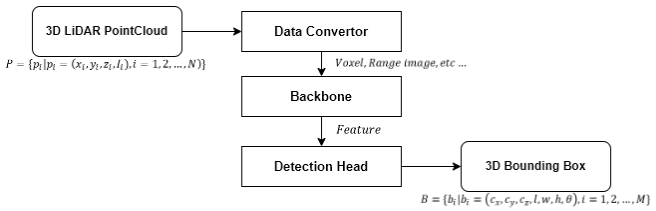

Fig. 2. **Structure of existing method** First, transform the data if necessary. Next, generate features through the backbone. Finally, predict the result through the head and extract the 3D bounding box.

detection relies solely on range view data. Experiments comparing voxel-based and range view-based methods using the aiMotive dataset[5] demonstrated over a five-fold performance improvement for objects beyond 50 meters.

The main contributions of this paper can be summarized as follows:

- A clustering-based method for vehicle detection, reducing computational complexity and providing a more efficient detection approach.
- Improved long-range object detection performance by using features specific to distant objects for classification.
- Demonstrated performance improvements for long-range object detection compared to existing methods through experiments with the aiMotive dataset[5].

II. VEHICLE DETECTION

A typical 3D LiDAR-based object detection method is illustrated in the Fig. 2. First, LiDAR data is input, and then converted into voxel representations. Afterward, features are extracted through a backbone network, and finally, a detection head generates the 3D bounding boxes. Such frameworks are essential for accurately detecting object positions and orientations. However, this framework is not suitable for racing environments, where minimizing computational time and enhancing detection range are critical.

To address the above issues, this paper proposes a clustering-based framework, as shown in Fig. 3, instead of the conventional deep learning-based framework. The proposed framework receives LiDAR data as input and removes the ground data and foreground data through a preprocessing step. Then, clustering is performed to create clusters, and features are extracted for each cluster. These features are used with AdaBoost to detect vehicle objects. Finally, based on the detected objects, 2D bounding boxes are generated in the range view.

A. Preprocessing

1) Ground Removal: LiDAR data consists of ground and object data, with ground data occupying a larger portion. This ground data not only interferes with object detection but also results in unnecessary computational time. To address this issue, this paper performs ground removal.

The LiDAR data is input in Cartesian coordinates, consisting of $(x, y, z, intensity)$. Before performing ground removal, this paper transforms the Cartesian coordinate data into a polar coordinate-based grid. The transformation process is described by Equation (1).

$$\begin{aligned} r &= \sqrt{x^2 + y^2}, \\ \theta &= \arctan\left(\frac{y}{x}\right) \end{aligned} \tag{1}$$

The grid consists of (r, θ), where each grid cell stores the LiDAR data corresponding to the respective (r, θ) values.

Subsequently, ground removal is performed based on whether each grid is solely composed of ground data. The determination of ground removal is based on the difference in the z values between neighboring grids, where the z values refers to the maximum z values within a grid.

Grids located at the same θ are sorted in ascending order with respect to r. Then, the z values of the grids within the same θ are compared, and if the difference exceeds a threshold, it is assumed that the grid contains object data in addition to ground. By scanning all grids, the data from grids containing only ground data are removed as shown in Fig. 4 (b), and the remaining object data are passed on to the next step.

2) Filtering & Downsampling: Clustering often suffers from increased computational time as the amount of data grows. To address this issue, this paper introduces Filtering and Downsampling steps to remove unnecessary data.

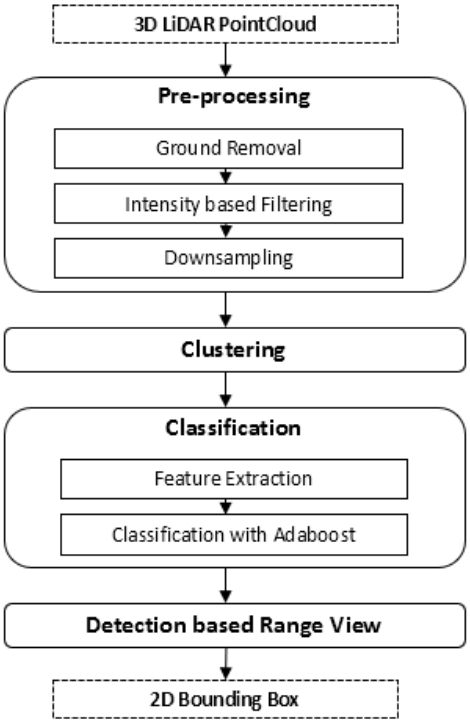

Fig. 3. **Structure of proposed method** 3D LiDAR data is provided as input and processed through a Pre-processing stage to remove ground and foreground data. The remaining data is then clustered to generate object candidates, and classification is performed to identify vehicle objects. Finally, detection results are extracted using a range view based detection method.

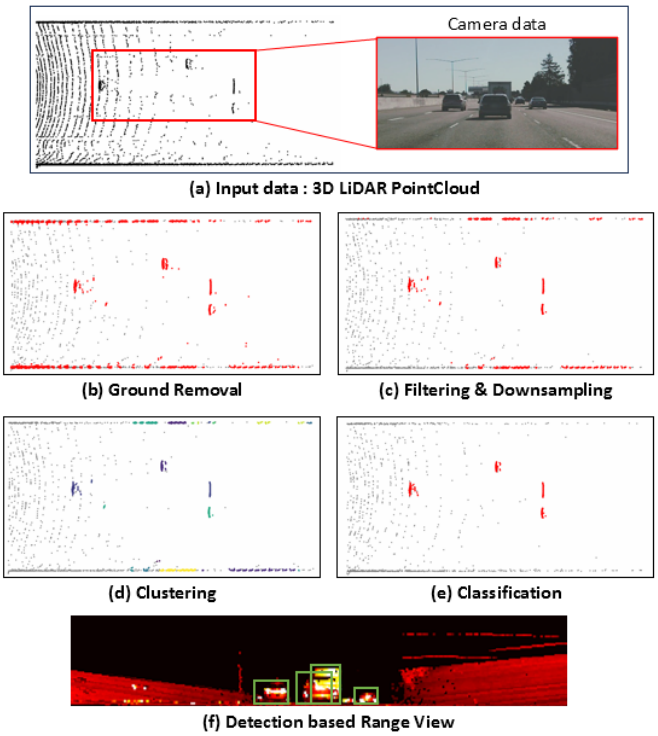

Fig. 4. **Process results of proposed method** (a) Input data : 3D LiDAR pointcloud. (b) Ground Removal. (c) Filtering & Downsampling. (d) Clustering. (e) Classification. (f) Detection based Range View

Intensity-based filtering is a process applied to data extracted through ground removal, where non-vehicle data is eliminated based on intensity values. Vehicles in LiDAR data typically exhibit higher intensity values compared to general objects due to their material properties. Leveraging this characteristic, this paper removes data below a certain intensity threshold to filter out foreground data.

Following the filtering step, a downsampling process is applied to further reduce the data size. LiDAR data inherently contains more points for objects closer to the sensor, leading to an increased amount of data. This increase contributes to higher computational time in subsequent processing steps. To mitigate this issue, this paper applies additional downsampling to significantly reduce the data size while preserving essential information. The results of this process are shown in Fig. 4 (c).

B. Clustering

After performing preprocessing stage, only object data is inputted. In this step, clustering is performed on the object data to generate object candidates, which are then used as input for the subsequent classification stage. Clustering is performed using Euclidean clustering, which is commonly employed in most methods, to extract object C_k. The results extracted through clustering can be distinguished by color, as shown in the Fig. 4 (d).

C. Classification

1) Feature Extraction: Feature extraction for vehicle classification is essential for each object. This section explains the features used in this paper and the rationale behind their selection. Feature extraction is performed for each object C_k, and the features used in this paper are given by Equation (2).

$$\mathbf{F}_{C_k} = \{d_k, (l_k, w_k, h_k), N_k, I_k, (vx_k, vy_k, vz_k), (rx_k, ry_k, rz_k), vd_k\} \quad (2)$$

The following features are computed for each cluster C_k:

- d_k: The average distance from the LiDAR sensor to the points in cluster C_k. This can be expressed as Equation (3).

$$d_k = \frac{1}{N_k} \sum_{i=1}^{N_k} \sqrt{(x_i)^2 + (y_i)^2 + (z_i)^2} \quad (3)$$

  x_i, y_i, z_i represent the x, y, and z coordinates of the LiDAR data belonging to cluster C_k.
- (l_k, w_k, h_k): The length, width, and height of the cluster, representing the spatial dimensions of the cluster in the point cloud.
- N_k: The number of points in cluster C_k.
- I_k: The average intensity of the points in cluster C_k. It is calculated as shown in Equation (4).

$$I_k = \frac{1}{N_k} \sum_{i=1}^{N_k} I_i \quad (4)$$

  where I_i is the intensity of the LiDAR data in cluster C_k.
- (vx_k, vy_k, vz_k): The variance of the point cloud in the x-, y-, and z-axes for cluster C_k. This can be expressed as Equation (5).

$$vx_k = \frac{1}{N_k} \sum_{i=1}^{N_k} (x_i - \bar{x}_k)^2 \quad (5)$$

  $\bar{x}_k$ represents the mean x value of the LiDAR data in cluster C_k, and vy_k, vz_k are also calculated using the equation (5) above.
- (rx_k, ry_k, rz_k): The variance of the point cloud compared to the sensor's resolution in the x-, y-, and z-axes, indicating the spread of data relative to the sensor resolution. This is calculated by examining the difference between the ideal variance based on the LiDAR sensor's resolution and the actual variance observed in the data.
- vd_k: The variance of the distance values within the cluster, representing how much the distance of points within the cluster varies from the average distance d_k. This can be expressed as Equation (6).

$$vd_k = \frac{1}{N_k} \sum_{i=1}^{N_k} (d_i - d_k)^2 \quad (6)$$

  where d_i is the distance of the LiDAR data in cluster C_k.

These 13 features are used as inputs for classification, and each feature serves as a distinctive representation that allows for a comparative differentiation between objects and vehicles.

2) Classification using Adaboost: The generated clusters are classified as either vehicles or non-vehicles using a machine learning approach. Inspired by [6], this paper employs the AdaBoost algorithm for classification. AdaBoost is a machine learning technique that combines multiple weak classifiers, iteratively creating new models that correct the weaknesses of the previous ones. Through this feature of compensating for weaknesses, the proposed method is capable of classifying even distant objects accurately.

The input to the AdaBoost classifier consists of the previously extracted features from each cluster. The final output is a binary classification, indicating whether the cluster represents a vehicle or not. This process allows the system to detect and isolate data corresponding to vehicle objects. The results of classification through this process are shown in the Fig. 4 (e)

D. Detection based Range View

Before generating the bounding box, the vehicle object data is transformed from the Cartesian coordinate system (x, y, z) to the spherical coordinate system (r, θ, ϕ). The transformation is expressed as shown in Equation (7).

$$\begin{aligned} r &= \sqrt{x^2 + y^2 + z^2}, \\ \theta &= \arctan\left(\frac{y}{x}\right), \\ \phi &= \arcsin\left(\frac{z}{r}\right) \end{aligned} \tag{7}$$

Using the coordinates (r, θ, ϕ), the range view image is generated through Equation (8).

$$\begin{aligned} \text{row} &= \left\lfloor \frac{\phi - \phi_{\min}}{\phi_{\max} - \phi_{\min}} \cdot H \right\rfloor, \\ \text{col} &= \left\lfloor \frac{\theta - \theta_{\min}}{\theta_{\max} - \theta_{\min}} \cdot W \right\rfloor \end{aligned} \tag{8}$$

H and W represent the height and width of the range view image. After extracting the top-left and bottom-right pixels of each object in the range view image, a 2D bounding box is generated. This 2D bounding box consists of the top-left pixel, bottom-right pixel, width, height, and center pixel. The final detection results are shown in Fig. 4 (f).

III. EXPERIMENTS

In this section, we describe the dataset, the implementation details, and compare and analyze the experimental results.

A. Dataset

This paper uses the aiMotive dataset[5] to evaluate long-range object detection. The aiMotive dataset[5] is a multi-sensor, long-range dataset that employs a 64-channel spinning LiDAR capable of detecting objects up to 200 meters. The dataset is categorized into scenes such as highways, urban environments, nighttime, and rainy conditions. In this paper, experiments are conducted using the highway scene from the high-speed environment subset. The highway scene is distributed across various distances, as shown in the Table I, and the detection performance is evaluated based on distances of 0-50m, 50-100m, and 100-150m.

TABLE I
ANALYSIS OF AIMOTIVE DATASET DISTRIBUTION FOR DISTANCE

Distance [m]	0 - 50	50 - 100	100 - 150	150 - 200
Frame Number	8204	8368	4737	55

TABLE II
EVALUATION FOR PERFORMANCE

Method	**Vehicle mAP@50 [%]**		
	0 - 50 [m]	50 - 100 [m]	100 - 150 [m]
HEDNet[3]	76.37	9.09	-
VoxelNeXt[4]	81.37	9.09	1.82
RangeRcnn[7]	**84.13**	-	-
Proposed	79.8	**90.3**	**82.2**

B. Implementation Details

First, in the ground removal step, the Cartesian coordinates are transformed into polar coordinates to create a grid, where the grid is divided based on θ=0.1 and r=0.5. The difference between neighboring grids is used to determine the presence of objects, with a threshold set at 0.2. During the filtering process, data within 50 meters with an intensity value lower than 30 was retained, while for data beyond 50 meters, only those with an intensity value greater than 30 were retained. Afterward, downsampling was performed with a threshold of 0.4. For the clustering process, the distance parameter is set to 1.0 to extract all clusters, and the minimum number of points parameter is set to 2. Lastly, classification is performed using AdaBoost. The input features for the classifier consist of 13 features of the clusters as described in Section 2.3, and the number of boosting rounds, T, is set to 50.

C. Evaluation

In this section, a comparative evaluation was conducted using the aiMotive dataset[5] to assess the proposed method against existing methods. Three baseline methods were selected for comparison: one that improves performance by converting sparse features into dense features[3], and another that enhances performance by directly utilizing sparse features[4], and a third based on range view-based detection methods[7]. To evaluate performance, the mean Average Precision (mAP) of 2D bounding boxes at different distances was used as the performance metric. The experimental results are shown in the Table II and the Fig. 5

The proposed method was compared with existing methods, and the results showed similar performance to the baseline for short-range objects within 50m. However, for long-range objects beyond 50m, the proposed method achieved

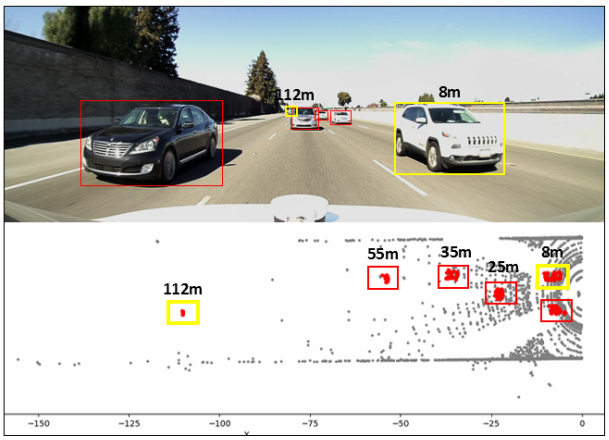

Fig. 5. **Qualitative result of proposed method** Vehicles were detected from 8m to 112m.

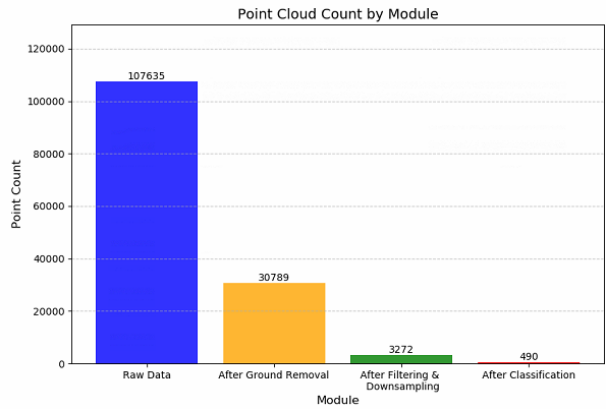

Fig. 6. **Point Cloud Count Analysis of LiDAR Data by Module Execution Results**

approximately an 80% improvement in performance while maintaining the performance for short-range objects.

Existing methods demonstrated high mAP scores of about 80% within 50m but showed a significant decrease in performance, down to approximately 10%, beyond 50m. This performance degradation was attributed to the sparsity of LiDAR data at greater distances, leading to inconsistent feature extraction for the same vehicle and difficulties in optimizing a single model for all distances. Additionally, for long-range objects, the detection performance in range view-based methods decreases due to occlusion caused by short-range objects.

In contrast, the proposed method maintained high detection performance by selecting all potential vehicle candidates through clustering. Furthermore, by employing a machine learning-based vehicle classification, non-vehicle objects were effectively filtered out, resulting in robust detection performance even at longer distances. As a result, the proposed method achieved an mAP of approximately 80% across most distance ranges, with a significant improvement in long-range object detection.

TABLE III
EVALUATION FOR COMPUTATION TIME

Method	**FPS [hz]**
HEDNet[3]	15
VoxelNeXt[4]	16
RangeRcnn[7]	22
Proposed	**53**

TABLE IV
COMPUTATION TIME AND POINTCLOUD NUMBER FOR MODULE

Module	Preprocessing	Clustering	Classification	**Total**
Time[ms]	14.3	3.4	1.0	**18.7**

D. Computation time

In this section, the computational time of the proposed method was evaluated and compared with existing methods using the aiMotive dataset[5]. The evaluation was conducted based on the processing time required for a single LiDAR scan. The results of the evaluation are summarized in Table III.

The experimental results demonstrate that the proposed method significantly reduces computational time to approciately 19 ms per LiDAR scan, a notable improvement over existing methods.

Existing methods, being deep learning-based, require extensive computations for all LiDAR data, limiting their ability to minimize processing time. In contrast, the proposed method employs preprocessing to eliminate unnecessary data before clustering, effectively reducing the computational load. This preprocessing step decreases the volume of data to be processed in shown the Fig. 6, subsequently minimizing computational overhead. Additionally, the ground removal process is performed using a grid-based approach, enabling more efficient computation. This optimization allows the proposed method to achieve the computational times presented in Table IV.

IV. CONCLUSIONS

In racing environments, rapid computation and extended detection range are critical. However, existing methods face challenges in improving detection range due to the sparse nature of LiDAR data and their reliance on deep learning-based detection frameworks. Furthermore, processing all data simultaneously increases unnecessary computations, making it difficult to optimize computational time. To address these issues, this paper proposes a clustering-based LiDAR detection method for long-range vehicle detection. The proposed method begins by performing preprocessing stage to eliminate unnecessary data, thereby reducing computational overhead. Next, clustering is performed to detect objects, and Adaboost based classifier is employed to classify vehicle objects. The classifier is specifically designed to focus on

the characteristics of long-range vehicles, ensuring robust performance even for distant objects. Finally, a range-view based detection is applied to the classified vehicle objects to generate 2D bounding boxes. The proposed method was evaluated using a long-range dataset, achieving approximately 80% performance for objects beyond 50m and reducing computation time to 19 ms per scan. These results demonstrate the suitability of the proposed approach for application in racing environments. Future research will focus on not only estimating the vehicle's position but also exploring methods for approximating its orientation. Additionally, the research will extend to multi-object classification for improved detection performance.

ACKNOWLEDGMENT

This work was supported by Innovative Human Resource Development for Local Intellectualization program through the Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT)(IITP-2025-RS-2020-II201462)

REFERENCES

[1] A. Khoche, L. P. Sánchez, N. Batool, S. S. Mansouri, and P. Jensfelt, "Towards long-range 3d object detection for autonomous vehicles," in *2024 IEEE Intelligent Vehicles Symposium (IV)*, pp. 2206–2212, 2024.

[2] L. Wang and B. Goldluecke, "Sparse-pointnet: See further in autonomous vehicles," *IEEE Robotics and Automation Letters*, vol. 6, pp. 7049–7056, 10 2021.

[3] G. Zhang, C. Junnan, G. Gao, J. Li, and X. Hu, "Hednet: A hierarchical encoder-decoder network for 3d object detection in point clouds," in *Advances in Neural Information Processing Systems* (A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, eds.), vol. 36, pp. 53076–53089, Curran Associates, Inc., 2023.

[4] Y. Chen, J. Liu, X. Zhang, X. Qi, and J. Jia, "Voxelnext: Fully sparse voxelnet for 3d object detection and tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21674–21683, June 2023.

[5] T. Matuszka, I. Barton, Ádám Butykai, P. Hajas, D. Kiss, D. Kovács, S. Kunsági-Máté, P. Lengyel, G. Németh, L. Pető, D. Ribli, D. Szeghy, S. Vajna, and B. Varga, "aimotive dataset: A multimodal dataset for robust autonomous driving with long-range perception," 11 2022.

[6] M. Yoshioka, N. Suganuma, K. Yoneda, and M. Aldibaja, "Real-time object classification for autonomous vehicle using lidar," in *2017 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*, pp. 210–211, 2017.

[7] Z. Liang, M. Zhang, Z. Zhang, X. Zhao, and S. Pu, "Rangercnn: Towards fast and accurate 3d object detection with range image representation," 9 2020.