# Enhancing Real-Time Multi-Object Tracking with Pillar-Based Models: Integration of Class Information and BiFPN Neck

Haneul Moon
Convergence Security for Automobile
Soonchunhyang University
Asan, South Korea
mhn1565@sch.ac.kr

Seongkeun Park
Department of Smart Automobile
Soonchunhyang University
Asan, South Korea
keiny@sch.ac.kr

***Abstract*—In recent years, center-based tracking has been the primary approach for multi-object tracking (MOT) due to its speed and accuracy. However, 3D MOT operates in much more complex environments than 2D, facing challenges such as occlusion, object deformation, and real-time processing difficulties. Addressing these challenges is critical for real-time applications such as autonomous driving, robotic vision, and surveillance systems, where accurate and fast tracking directly impacts safety and efficiency. For instance, in autonomous vehicles, poor object tracking performance can lead to accidents or incorrect route planning. Therefore, ensuring accurate object detection and tracking performance in real time is essential in these fields. To address this issue, this study proposes Enhancing Real-Time Multi-Object Tracking with Pillar-Based Models, which applies BiFPN instead of the conventional FPN to enhance feature extraction and integrates object class information into the velocity estimation head for more precise tracking. Evaluation on the nuScenes dataset demonstrates that the proposed method significantly improves 3D MOT performance, resulting in a 5.8% increase in AMOTA and a reduction of IDS by 133.**

***Keywords—3D MOT, Detection, PointCloud***

## I. Introduction

In this study, we propose a method to improve the performance of real-time 3D multi-object tracking (MOT) based on the Pillar Base architecture. Real-time 3D MOT plays a crucial role in various applications such as robotics, autonomous vehicles, and surveillance systems. The main challenges include occlusion, deformation, interference from similar objects, and the difficulties of real-time processing. To address these challenges, researchers have explored various real-time methods such as center-based tracking systems [3] [4] [6].

PointPillars [7] in the Pillar Base architecture aggregates 3D point clouds into 2D pseudo-images, enhancing both real-time performance and data representation. This allows for real-time 3D detection while utilizing existing 2D network architectures [11] [12] and methods. Previous research has employed a Track-by-Detection approach based on Pillar-Based detection [2] [5] [7]. However, this method heavily depends on detection performance, and errors in detection can negatively affect tracking performance, making it difficult to maintain consistency across frames. As a result, End-to-End (E2E) approaches [3] that incorporate tracking directly into the model have been explored, with CenterPoint [4] being one such model.

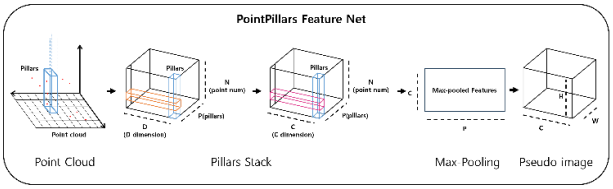


Figure 1. PointPillars Feature Net Structure.

CenterPoint [4] moves away from the traditional Anchor-based Bounding Box estimation method [5], using heatmap-based center point extraction [6] to detect objects more quickly and employing a velocity estimation branch that outperforms Kalman Filter-based trackers. However, the processing of 3D information and feature extraction pose a trade-off between real-time performance and accuracy, highlighting the need for methods that improve detection and tracking performance in real-time environments.

To address this issue, this study applies BiFPN [10] instead of the traditional FPN and integrates class information into the velocity estimation head in the Pillar Base architecture. This improves the efficiency of learning class-specific velocity estimation and enables more accurate tracking. The proposed method was evaluated using the nuScenes dataset [1], demonstrating enhanced performance in 3D multi-object tracking.

## II. Related Work

### *A. Pillar-Based Architecture for 3D Object Detection*

PointPillars architecture represents a major advancement in the field of 3D object detection. This method processes 3D point clouds by converting them into a 2D pseudo image [9], which allows for efficient and fast computation using 2D convolutional neural networks (CNNs). Unlike traditional 3D detection methods that rely on 3D convolutions [8, 14], PointPillars reduces computational complexity by transforming high-dimensional 3D data into a 2D form, greatly improving real-time performance.

The core innovation of Pillar-based architecture is the division of point cloud data into predefined pillars, which act as grid-like representations of the 3D space. These pillars are processed to extract features and then assembled into a pseudo-image that can be fed into a standard 2D CNN for further processing.

- Pillar-based division: Point cloud data is discretized into uniform pillars. Each pillar aggregates point

features, which are then processed to create a pseudo-image.

- 2D CNN processing: By converting 3D data into a 2D image, PointPillars leverages efficient 2D CNNs, significantly speeding up object detection compared to traditional 3D convolutional methods
- Real-time efficiency: This architecture drastically improves the processing time for 3D object detection, making it well-suited for applications like autonomous driving and robotic systems where real-time detection is crucial.

PointPillars has shown to be effective in improving the speed and accuracy of 3D object detection, making it a valuable approach in fields such as autonomous vehicles and robotics.

### B. *CenterTrack: 2D Object Tracking*

Tracking Objects as Points paper by Zhou et al. (2020) [3] introduces an innovative approach that integrates object detection and tracking. Unlike traditional anchor-based methods, this approach utilizes a heatmap to detect [6] and track the center point of each object. This allows for faster and more efficient object detection in 2D images, and, combined with velocity estimation, it outperforms classical methods such as Kalman Filters.

- Heatmap-based center point extraction: CenterTrack uses a heatmap to detect the center points of objects, bypassing the need for complex bounding box generation and enabling faster detection. This method also facilitates efficient tracking of object trajectories.
- Velocity estimation branch: CenterTrack includes a velocity estimation branch, which predicts the continuous positions of each object, improving tracking accuracy. It achieves superior performance compared to Kalman Filter-based trackers, especially in handling rapid changes in object velocity.
- End-to-End (E2E) approach: CenterTrack employs an End-to-End method, integrating object detection and tracking without the need for post-processing. This eliminates the need for separate prediction and update stages, maximizing tracking efficiency.

### C. *Center Point: 3D Object Tracking*

CenterTrack was initially developed for 2D object tracking, but with the advent of PointPillars [7], many 2D techniques could be adapted for 3D object detection. PointPillars converts 3D point clouds into 2D pseudo-images, making it easier to apply traditional 2D methods. This led to the development of models like CenterPoint, which is specifically designed for 3D environments.

CenterPoint builds on Center Track's principles, using PointPillars [7] and VoxelNet [14] to generate feature maps, and employs a heatmap-based approach to predict 3D object positions and velocities. By extending the strengths of 2D center-based tracking [3] to 3D, CenterPoint delivers strong performance in object detection and tracking.

- Heatmap-based Center Point Extraction: This technique enables efficient detection of object centers in 3D, eliminating the need for bounding box generation, which speeds up the process.

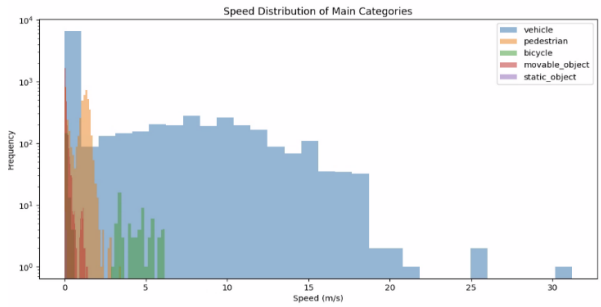

Figure 2. Speed Distribution of Main Categories.

- Velocity Estimation and 3D Tracking: CenterPoint includes a velocity estimation module, allowing it to predict and track objects' positions in real time [4].
- Performance and Speed Trade-off: Although PointPillars allows for real-time processing, it sacrifices some accuracy compared to VoxelNet. On the other hand, VoxelNet offers greater accuracy but lacks real-time processing capabilities.

To address this trade-off between speed and performance, we aim to enhance CenterPoint's performance by utilizing the real-time capabilities of PointPillars.

## III. Enhanced Tracking Methods

### A. *Class-Aware Velocity Estimation*

The speed distribution of major classes in the nuScenes dataset [1] shows varying ranges. For example, vehicles typically move between 0 and 23 km/h, while pedestrians range from 0 to 2.5 km/h. Bicycles and stationary objects show speeds between 2.5-7 km/h and 0-1.25 km/h, respectively, as shown in Fig 2. This distribution suggests that class information could potentially aid in estimating the speed of each object, forming the basis of our hypothesis.

To verify this hypothesis, we propose a method that utilizes class information for velocity estimation. First, a pseudo image is generated by the PointPillars Feature Net and transformed into a top-view format through mapview conversion. The transformed features are then passed through the head network, which performs regression analysis to estimate class heatmap, object size, sub-voxel location refinement, rotation, and velocity. Six feature maps, including reg, height, dim, rot, vel, and hm, are utilized for this process.

To incorporate class information into the velocity estimation head, the class information from the heatmap must be computed prior to the velocity head estimation. Once the class information is extracted from the heatmap, it is concatenated with the feature map used for velocity estimation. While the other feature maps contain 64-dimensional features, the velocity estimation feature map has 65 dimensions due to the added class information.

This method is structurally simple yet has proven effective in improving velocity estimation performance, as demonstrated by our experimental results. By integrating class information, we can predict the velocity of each object more accurately, leading to an overall enhancement in tracking performance.

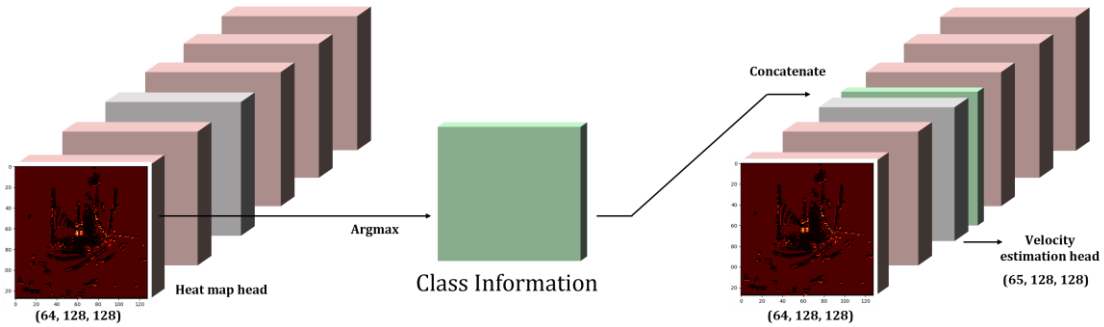


Figure 3. Concatenate the class information to the velocity estimation head

### B. *Optimizing Feature Pyramid for Better Object Tracking*

In object detection and tracking, the feature pyramid plays a crucial role in capturing multi-scale information to detect objects of various sizes. To improve the performance of the pyramid structure, there is a shift from the traditional FPN (Feature Pyramid Network) [14] to the more advanced BiFPN (Bi-directional Feature Pyramid Network) [10]. This optimization is key to improving accuracy in object detection and tracking tasks. The main improvement of BiFPN is the introduction of bi-directional information flow for more efficient processing of features. While traditional FPN processes features in a unidirectional manner, BiFPN restructures the flow with lateral connections, allowing information to flow both bottom-up and top-down. This results in more effective feature fusion across different scales.

Additionally, BiFPN introduces a learnable weighting mechanism that assigns importance to features at each pyramid level. This allows the network to emphasize more critical information, thereby enhancing detection and tracking performance by focusing on finer object details. Furthermore, nodes with only input edges, which do not contribute to feature fusion, are removed to optimize the structure.

In this study, we resized the Map-view transformed feature map (64, 512, 512) into P5 (64, 256, 256), P4 (128, 128, 128), and P3 (256, 64, 64) to serve as input for the BiFPN neck. Additionally, middle-connections were added to BiFPN to strengthen the direct connectivity between higher and lower layers. This increases the density of information fusion between layers, resulting in expected performance improvements. As a result, features from P3, P4, and P5 were fused to generate the final Feature Map (384, 128, 128).

By applying the structural advantages of BiFPN and its learnable weighting mechanism, we evaluated the potential improvements in accuracy and speed during the tracking process. The results confirmed performance improvements with an AMOTA increase of 1.7% and a reduction of IDS by 41.

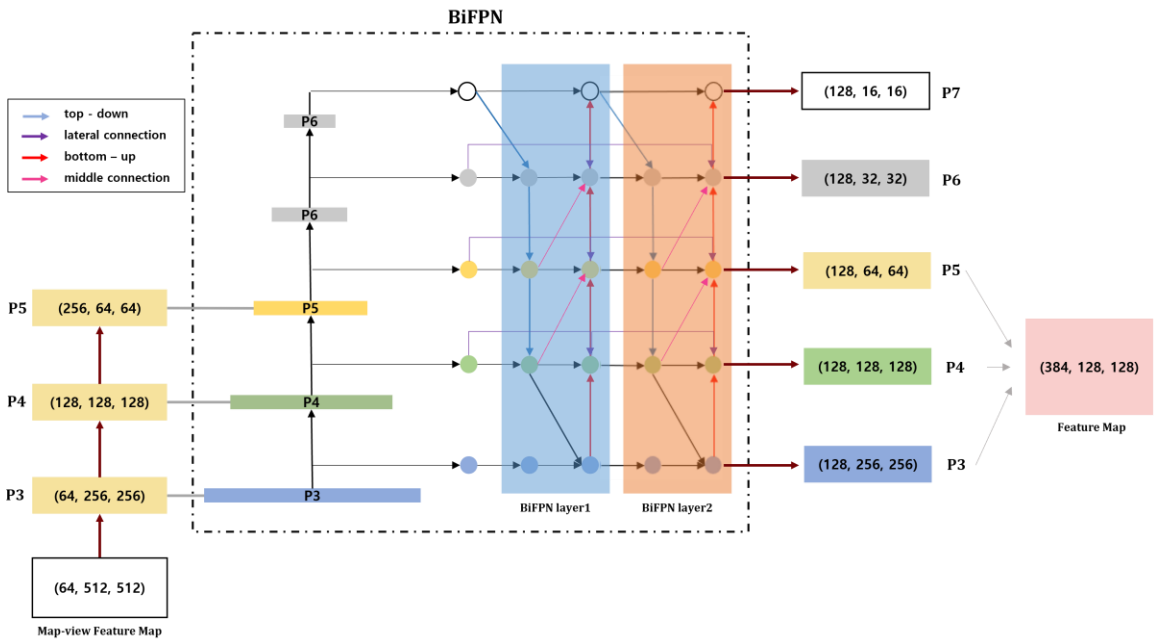


Figure 4. BiFPN with middle connections

## IV. Experiment And Result Analysis

### A. Experimental Setup

In this study, the experiments were conducted using the NuScenes dataset [1]. Out of a total of 850 data samples, 700 samples were used for training, while the remaining 150 samples were used for validation to assess model performance. For model performance evaluation, we applied the evaluation methodology provided by the vel dataset in NuScenes. The experiments were carried out on the following system configuration. The CPU used was an Intel(R) Core (TM) i9-10940X @ 3.30GHz, with a total of 62GB of memory and 15GB of swap memory. Additionally, two NVIDIA GeForce RTX 3080 Ti GPUs were employed to support high-speed parallel processing. The operating system was Ubuntu 18.04, and the programming language used for the experiments was Python 3.8, with CUDA version 11.8 to facilitate GPU acceleration. For validation, the performance was measured based on the official evaluation metrics of NuScenes. The key performance metrics included AMOTA (Average Multi-Object Tracking Accuracy), AMOTP (Average Multi-Object Tracking Precision), Recall, MOTAR (Multi Object Tracking Accuracy with Recovery), MOTA (Multi-Object Tracking Accuracy), MOTP (Multi-Object Tracking Precision), MT (Mostly Tracked), ML (Mostly Lost), FAF (False Alarms per Frame), GT (Ground Truth), TP (True Positives), FP (False Positives), FN (False Negatives), IDs (Identity Switches), Frag (Fragmentations), TID (Track Identification Error), and LGD (Log-Grounded Detection).

The performance of the model for multi-object tracking was comprehensively evaluated through these metrics, with particular focus on the accuracy of speed and position estimation. Each metric played a critical role in quantitatively analyzing the tracking and detection performance of the model, and the experiments confirmed improvements in the model's performance.

The model was trained using the Adam Optimizer, and a One Cycle Learning Rate Scheduler was employed for learning rate scheduling. The initial learning rate was set to 0.001, with the overall learning rate change controlled by div factor=10.0. During the first 40% of the training, the learning rate was increased, followed by a gradual decrease in the latter stages. The momentum parameters were set to moms= [0.95, 0.85], ensuring stable training and faster convergence. Additionally, Gradient Clipping was applied during training, with max norm=35 to prevent gradient explosion, and clipping was performed using L2 normalization with norm type=2.

TABLE I. Experimental Computer Setting

| *Category* | *Model* |
|---|---|
| CPU | Intel(R) Core (TM) i9-10940X CPU 3.30GHz |
| Memory | 62GB |
| Swap Memory | 15GB |
| GPU | NVIDIA GeForce RTX 3080 Ti * 2 |
| OS | Ubuntu 18.04 |
| Program Language | Python 3.8 |
| CUDA Version | 11.8 |

### B. Performance Evaluation

The performance of the proposed model was evaluated using 150 validation scenes from the NuScenes dataset [1], comparing it against the original CenterPoint [4] model and a Kalman Filter-based tracking system [5]. The baseline model using a pillar-based FPN achieved an AMOTA of 54.2% with 694 IDS occurrences. The model incorporating class information fusion, proposed in this study, demonstrated an improvement with a 4.1% increase in AMOTA and a reduction of 92 IDS occurrences. Furthermore, the model integrating both class information and a BiFPN Neck showed a 1.7% increase in AMOTA and a decrease of 59 IDS occurrences. This translates to a total performance improvement of 5.8% in AMOTA and a reduction of 133 IDS occurrences compared to the original CenterPoint pillar-based model.

The results of the class-wise AMOTA performance evaluation assessed the models' performance across various object classes, including Bicycle, Bus, Car, Motorcycle, Pedestrian, Trailer, and Truck. The AMOTA metric provides a comprehensive measure of accuracy in tracking multiple

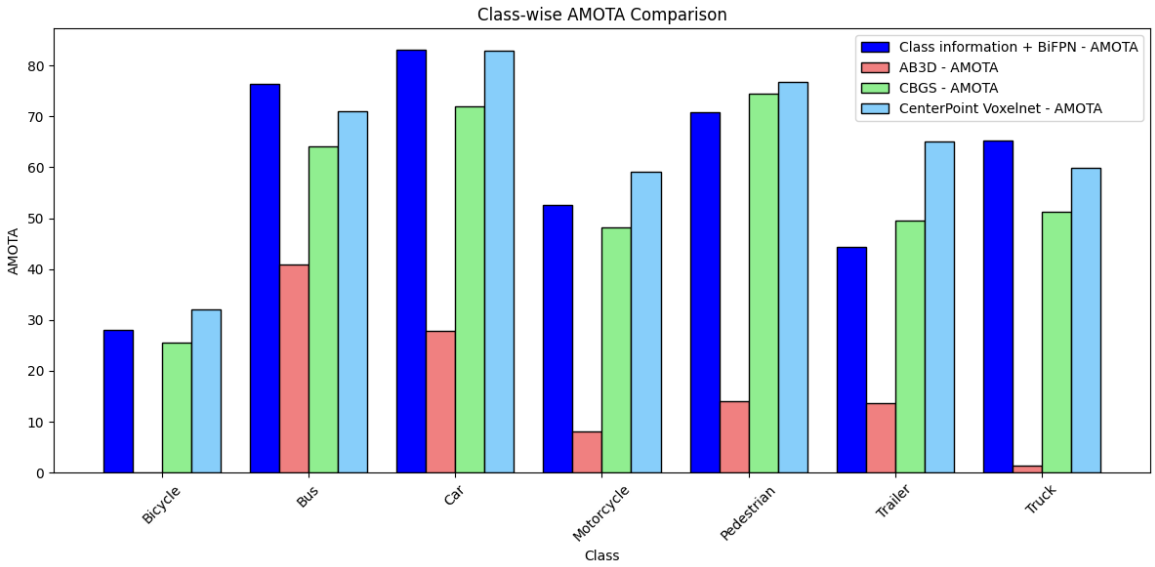


Figure 5. This figure shows the class-wise comparison of AMOTA (Average Multi-Object Tracking Accuracy) across four models. The compared models are: (1) the proposed model with integrated class information and BiFPN (blue), (2) the AB3D model (red), (3) the CBGS model (green), and (4) the CenterPoint VoxelNet model (light blue).

TABLE II. PERFORMANCE COMPARISON ON NUSCENES VAL

| *Method* | *AMOTA* | *IDS* | *Total time* |
|---|---|---|---|
| PointPillars + Class info + BiFPN + CenterTrack | 60.0 | 561 | 40ms |
| PointPillars + Class info + FPN + CenterTrack | 58.3 | 602 | 36ms |
| PointPillars + FPN + CenterTrack | 54.2 | 694 | 36ms |
| VoxelNet + FPN + CenterTrack | 62.6 | 822 | 76ms |
| PointPillars + FPN + Mahalanobis Kalman Filter(CBGS) | 57.0 | 900 | 282ms |

objects simultaneously, indicating how well each model tracks objects across different classes.

The proposed model exhibited superior performance in most classes, particularly in the Bus, Car, and Truck classes, where it recorded higher AMOTA values compared to other models. The significant performance gain in the Bus and Car classes can be attributed to the BiFPN neck ability to effectively fuse multi-scale features of large objects and the integration of class-specific velocity information, which aligns well with the characteristics of these classes in the dataset, such as their high speed and prominent size. The performance on small objects is still inferior to VoxelNet, likely due to the influence of the pillar-based architecture's pillar size, which poses a limitation. The CBGS [15] model showed competitive performance in the Car, Bus, and Truck classes, but still lagged the proposed model.

The CenterPoint VoxelNet model demonstrated overall superior performance; however, the proposed model excelled particularly in tracking large or fast-moving objects such as trucks, buses, and cars. This comparison highlights that the proposed model has a significant advantage in specific classes while maintaining competitive performance across other classes.

## V. CONCLUSIONS

In this study, we proposed a novel approach to improving 3D multi-object tracking performance by integrating class information into velocity estimation and employing a BiFPN neck architecture. The proposed model was evaluated using the NuScenes dataset, showing significant improvements over the original CenterPoint model and a Kalman Filter-based tracking system. Specifically, it achieved a 5.8% increase in AMOTA and a reduction of 133 IDS occurrences compared to the pillar-based CenterPoint model. By incorporating class information fusion and the BiFPN neck, the model not only enhanced tracking accuracy but also reduced the occurrence of ID switches. Moreover, the addition of the BiFPN architecture substantially improved feature extraction capabilities, thereby positively influencing overall tracking performance. Although this improvement resulted in a slight increase in processing time (40 ms per frame), it remains a promising approach for real-time applications where both accuracy and reliability are paramount.

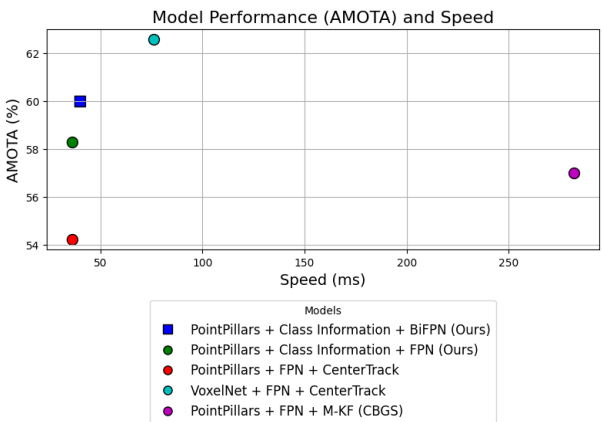


Figure 6. Model Performance and Speed, this figure presents the comparison of speed and performance (AMOTA) across various models. Each model is represented by a different color, with the proposed model (PointPillars + Class Information + BiFPN) highlighted using a square marker.

Overall, the results of this study demonstrate that integrating class information and an advanced feature pyramid architecture effectively enhances both the accuracy and robustness of 3D multi-object tracking systems.

For future work, we plan to utilize optimization frameworks such as ONNX and TensorRT, along with other model compression techniques, to further boost real-time performance and scalability. By reducing computational overhead and memory usage, we aim to maintain robustness under various environmental conditions (e.g., weather, lighting) and explore possibilities for handling large-scale datasets. In addition, we intend to investigate more advanced fusion methods and conduct evaluations on diverse datasets to generalize our model's performance across a wider range of scenarios.

## ACKNOWLEDGMENT

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (RS-2023-00245084), Institute for Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2022-0-01197, Convergence security core talent training business (Soonchunhyang University))

## REFERENCES

[1] 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: CVPR (2020)

[2] Bewley, A., Ge, Z., Ott, L., Ramos, F., & Upcroft, B. (2016). Simple online and realtime tracking. In Proceedings of the IEEE International

[3] Zhou, X., Koltun, V., & Krähenbühl, P. (2020). Tracking Objects as Points. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 474-490). UT Austin, Intel Labs.

[4] Yin, T., Zhou, X., & Krähenbühl, P. (2021). Center-based 3D Object Detection and Tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 11784-11793). UT Austin.

[5] Weng, X., Wang, J., Held, D., & Kitani, K. (2020). AB3DMOT: A Baseline for 3D Multi-Object Tracking and New Evaluation Metrics. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 10438-10445). Robotics Institute, Carnegie Mellon University.

[6] Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., & Tian, Q. (2019). CenterNet: Keypoint Triplets for Object Detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (pp. 270-279). University of Chinese Academy of Sciences, University of Oxford, Huawei Noah's Ark Lab.

[7] Lang, A. H., Vora, S., Caesar, H., Zhou, L., Yang, J., & Beijbom, O. (2019). PointPillars: Fast Encoders for Object Detection from PointClouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 12697-12705). nuTonomy: an APTIV company.

[8] Yan, Y., Mao, Y., & Li, B. (2018). SECOND: Sparsely Embedded Convolutional Detection. In Sensors (Vol. 18, No. 10, pp. 3337-3348). Chongqing University, TrunkTech Co., Ltd.

[9] Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). PointNet: Deep learning on point sets for 3D classification and segmentation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 652-660. https://doi.org/10.1109/CVPR.2017.16

[10] Tan, M., Pang, R., & Le, Q. V. (2020). EfficientDet: Scalable and Efficient Object Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 10781-10790). Google Research, Brain Team

[11] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 39(6), 1137-1149.

[12] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 770-778).

[13] Zhou, Y., Tuzel, O. (2018). VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 4490-4499).

[14] Lin, T. Y., Dollar, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature Pyramid Networks for Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 2117-2125). Facebook AI Research (FAIR), Cornell University.

[15] Zhu, B., Jiang, Z., Zhou, X., Li, Z., & Yu, G. (Year). Class-balanced Grouping and Sampling for Point Cloud 3D Object Detection. Megvii Research, Institute of Automation, Chinese Academy of Sciences, Tsinghua