*electronics*

MDPI

*Article*

# Multi-Scale Feature Fusion Point Cloud Object Detection Based on Original Point Cloud and Projection

**Zhikang Zhang [1,2], Zhongjie Zhu [1,2,*], Yongqiang Bai [2], Yiwen Jin [2] and Ming Wang [1,2]**

[1] Faculty of Information Science and Engineering, Ocean University of China, Qingdao 266100, China; zhangzhikang@stu.ouc.edu.cn (Z.Z.); acwarming@stu.ouc.edu.cn (M.W.)
[2] Ningbo Key Lab of DSP, Zhejiang Wanli University, Ningbo 315100, China; yongqiangbai@zwu.edu.cn (Y.B.); 2022883025@zwu.edu.cn (Y.J.)
* Correspondence: zhuzhongjie@zwu.edu.cn

**Abstract:** Existing point cloud object detection algorithms struggle to effectively capture spatial features across different scales, often resulting in inadequate responses to changes in object size and limited feature extraction capabilities, thereby affecting detection accuracy. To solve this problem, we present a point cloud object detection method based on multi-scale feature fusion of the original point cloud and projection, which aims to improve the multi-scale performance and completeness of feature extraction in point cloud object detection. First, we designed a 3D feature extraction module based on the 3D Swin Transformer. This module pre-processes the point cloud using a 3D Patch Partition approach and employs a self-attention mechanism within a 3D sliding window, along with a downsampling strategy, to effectively extract features at different scales. At the same time, we convert the 3D point cloud to a 2D image using projection technology and extract 2D features using the Swin Transformer. A 2D/3D feature fusion module is then built to integrate 2D and 3D features at the channel level through point-by-point addition and vector concatenation to improve feature completeness. Finally, the integrated feature maps are fed into the detection head to facilitate efficient object detection. Experimental results show that our method has improved the average precision of vehicle detection by 1.01% on the KITTI dataset over three levels of difficulty compared to Voxel-RCNN. In addition, visualization analyses show that our proposed algorithm also exhibits superior performance in object detection.

**Keywords:** point object detection; 3D swin transformer; 2D/3D feature fusion

check for updates

**Citation:** Zhang, Z.; Zhu, Z.; Bai, Y.; Jin, Y.; Wang, M. Multi-Scale Feature Fusion Point Cloud Object Detection Based on Original Point Cloud and Projection. *Electronics* **2024**, *13*, 2213. https://doi.org/10.3390/electronics13112213

Academic Editors: Silvia Liberata Ullo and Li Zhang

Received: 12 May 2024
Revised: 30 May 2024
Accepted: 2 June 2024
Published: 6 June 2024

**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

## 1. Introduction

Point cloud object detection, an essential technology for autonomous driving, robotics and medical imaging, is attracting considerable attention. This technique has established itself at the forefront of artificial intelligence and computer vision research, attracting the investigative interest of many researchers. However, the inherent characteristics of point cloud data, which include sparsity, irregularity, and disorder [1], amplify the difficulty of object recognition, especially compared to 2D image detection, and its technical implementation faces more severe challenges. Processing 3D point cloud data typically requires a large amount of computational resources. Methods such as 3D convolutional and graph neural networks can effectively process 3D data, but their computational complexity is much higher than that of 2D convolutional neural networks, requiring higher hardware equipment and slower processing speeds. Point cloud data acquired by sensors (e.g., LiDAR, RGB-D cameras) can be affected by environmental conditions, producing more noise points and incomplete targets, which poses a major challenge to the accuracy of object detection.

In addressing point cloud object detection tasks, deep learning methods are widely applied for feature extraction and classification tasks. Similarly, Zheng et al. [2] proposed a

manifold regularization-based deep convolutional autoencoder (MR-DCAE) for unauthorized broadcasting identification, demonstrating the potential of deep learning in complex signal recognition. Existing methods for point cloud object detection can be categorized into three distinct categories: projection-based techniques, which map the data onto a 2D plane; voxelization-based methods, which convert the data into a 3D grid of volumetric elements; and direct point cloud-based approaches, which process the data in its native, unaltered form. The point cloud projection-based approach projects the point cloud into 2D views first, uses a mature 2D object recognition network to identify the object position, and then find the corresponding point cloud object through 3D remapping operation to achieve point object detection. In 2016, Li et al. [3] presented the 3D Fully Convolutional Network (FCN) network, an innovative adaptation of the 2D FCN framework for use in point cloud object detection. This method leverages upsampling strategies to reinstate the dimensions of images, thereby improving detection accuracy. Beltran et al. [4] developed the BirdNet network, which uses Faster R-CNN to process the bird's eye view (BEV) of a point cloud, facilitating the localisation of targets on a 2D plane and the extraction of their 3D attributes. In 2019, Yang et al. [5] created the PIXOR network, which converts point cloud data into 2D BEVs to perform single-stage predictions at the pixel level. Each prediction directly corresponds to a 3D object estimate, using advanced 2D convolutional techniques to achieve effective 3D object detection.

Unlike projection-based approaches, voxel-based organize unordered point clouds into structured voxel grids and use 3D Convolutional Neural Networks (CNNs) to extract features for object detection. Zhou et al. [6] introduced the VoxelNet network, which uses the Voxel Feature Encoding (VFE) layer for point cloud encoding. Yan et al. [7] introduced the SECOND network, which increases training speed by using sparse 3D convolutions and improves orientation estimation accuracy with a directional angular regression loss. In 2019, Wang et al. [8] introduced the Voxel-FPN network, which uses a bottom-up dense aggregation structure to encode multiscale voxel features, significantly simplifying the network architecture and improving detection performance.

However, both projection-based and voxelization-based point cloud detection methods have inherent limitations. Specifically, methods based on projection organize the 3D point cloud into a 2D BEV using a projection matrix, inevitably sacrificing feature information from the height dimension, thereby reducing detection accuracy. Meanwhile, voxelization approaches lead to information loss when the conversion process involves a number of points and voxels that surpass the set capacity threshold, which adversely affects detection performance. To overcome the above mentioned drawbacks, researchers have developed methods that directly use the original point cloud for object detection. In 2016, Qi et al. [9] introduced the PointNet algorithm, integrating spatial transformation networks and max-pooling operations to tackle the issues of rotational invariance and the disordered nature of point clouds, thereby enabling efficient object detection. Subsequently, Qi et al. [10] proposed PointNet++, improving model performance through a multi-level feature extraction framework that uses farthest point sampling (FPS) to select pivotal points, collects k-nearest neighbors around these points, and extracts features through the core network, significantly improving detection accuracy. In 2020, Yang et al. [11] proposed the 3DSSD method, which introduced a sampling fusion strategy and an anchor-free regression head, along with a 3D center-ness assignment strategy, further refining the accuracy and efficiency of detection. Although these methods skillfully preserve the object features in 3D space, they remain deficient in capturing spatial positional features at different scales and in responding to changes in target size, resulting in object detection accuracy that needs to be improved.

Therefore, we propose a point cloud object detection method that employs a multi-scale feature fusion strategy that uses both raw point cloud data and its projection. Multiscale feature extraction and fusion play a critical role in addressing complex point cloud object detection problems. These methods not only capture features at different scales but also effectively handle variations in object size and detection challenges in complex environ-

ments. He et al. [12] proposed an adaptive global–local generalized finite element method for solving multiscale advection–diffusion problems, Zhang et al. [13] proposed a fine-grained modulation classification method based on a multi-scale radio transformer with dual-channel representation, demonstrating significant advantages of multiscale methods in improving computational accuracy and efficiency. This method skillfully captures and preserves spatial location features across different scales, enhancing the performance of point cloud object detection. The primary contributions of the paper can be summarized as follows:

- We have developed a feature extraction module based on the 3D Swin Transformer [14], which uses 3D patch partition technology for point cloud preprocessing. This module integrates a 3D sliding window mechanism to implement self-attention and capture and extract spatial features at multiple scales through a systematic downsampling process. This approach not only improves the model's ability to capture 3D spatial features, but also increases both processing efficiency and feature extraction accuracy. We also built a 2D feature extraction module that uses projection techniques to convert 3D point clouds into 2D images and uses Swin Transformer to extract 2D features.
- We have developed an innovative 2D/3D feature fusion module that fuses 2D and 3D features at the channel level using point addition and vector concatenation techniques. This fusion strategy not only increases the integrity and expressiveness of the features, but also improves the model's ability to detect targets in complex environments.

## 2. Proposed Algorithm

As shown in Figure 1, the method described uses a 3D Swin Transformer for multi-scale feature fusion in point cloud target detection. It is composed of three primary components: a 3D feature extraction module based on the 3D Swin Transformer, a feature projection and 2D feature extraction module and a 2D/3D feature fusion module. Firstly, the point cloud is processed through a 3D Swin Transformer-based feature extraction module and downsampling to obtain 3D features across different scales. At the same time, a Swin Transformer is used to extract 2D features from the point cloud by projecting it onto a 2D plane. These 2D and 3D features are then input into the 2D/3D feature fusion module, where they are integrated at the channel level by element-wise addition and vector concatenation to form enhanced 3D features. Finally, these integrated 3D features are input into the detection head, where the object detection framework is established through an RPN network and ROI pooling module, completing the point cloud object detection process.

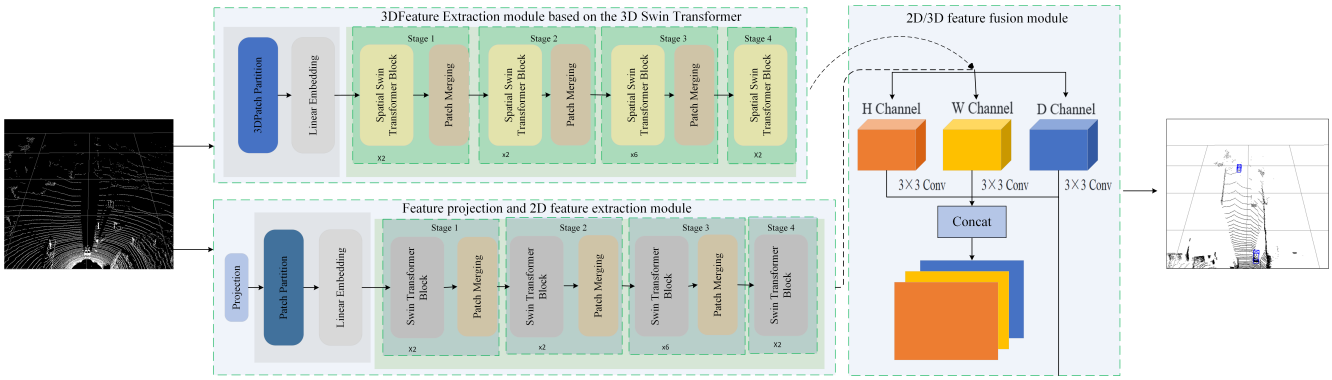



**Figure 1.** An overview of our methodology.

### *2.1. The 3D Feature Extraction Module Based on the 3D Swin Transformer*

The 3D feature extraction module based on the 3D Swin Transformer consists of data preprocessing and feature extraction modules, as shown in Figure 1. First, the original point cloud $F$ with size $D \times H \times W \times 3$ is inputted into the preprocessing module. The selection of a specific size $D \times H \times W \times 3$ for the original point cloud aims to standardize

the input data. The initial point cloud is segmented into data block $P^{l-1}$ with size $2 \times 4 \times 4$ by 3D segmentation layers. Segmenting the original point cloud into smaller blocks allows for finer-grained feature extraction, capturing local spatial features more effectively. Next, linear embedding is applied to transform the data in channels $D$, $H$ and $W$, and then, normalization is performed along the feature dimension, and the dimensionality of the data block $P^{l-1}$ is altered to $D/2 \times H/4 \times W/4 \times C$. The linear embedding layer and normalization operations can reduce the dimensionality of the data while increasing the number of feature channels, thus enhancing the model's representation of features and processing efficiency.

The data block $P^{l-1}$ is then input to the Spatial Swin Transformer module to extract the feature, the structure of which is shown in Figure 2, including four LN layers, one 3D W-MSA layer, one 3D SW-MSA layer, two MLP layers, and four residual connection layers.

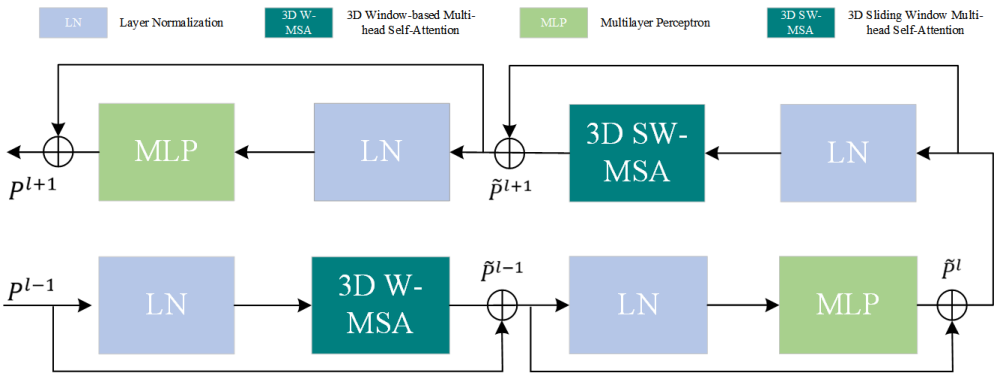



**Figure 2.** Spatial Swin Transformer module.

To effectively extract spatial features located at different positions, we employ the 3D Window-based Multi-head Self-Attention layer (3D W-MSA) within the Spatial Swin Transformer module. Initially, the input data block $P^{l-1}$ is uniformly divided into multiple 3D windows of size $N \times M \times M$ (where $N$ represents depth, and $M$ represents both height and width), ensuring these windows do not overlap. Subsequently, each 3D window is further partitioned into $h$ subspaces on the feature dimension, where $h$ denotes the number of attention heads. Each subspace undergoes linear transformations for each feature vector's Query ($Q$), Key ($K$), and Value ($V$) through three distinct parameter matrices $W^Q$, $W^K$, and $W^V$. Following this, scaled dot-product self-attention computations are performed within each subspace, and the outcomes from the $h$ attention heads are combined and fused using a learnable weight matrix to produce the multi-head self-attention features $\tilde{P}^{l-1}$; this allows feature information from many subspaces to be obtained. The scaled dot-product attention computation formula for the $n$th attention head is as follows:

$$h_n = \text{Attn}(QW_n^Q, KW_n^K, VW_n^V) \tag{1}$$

where $h_n$ represents the attention computation result for the $n$th attention head scaled dot-product method, and $W_n^Q, W_n^K, W_n^V$, respectively, signify the $n$th matrix parameters $W^Q, W^K$, and $W^V$. Attn represents the scaled dot-product model and is calculated as follows:

$$\text{Attn}(Q, K, V) = \text{SoftMax}\left(\frac{QK^T}{\sqrt{d_k}} + C\right)V \tag{2}$$

where $QK^T$ denotes the mechanism of information exchange among different features, utilizing the dot product to ascertain the similarity between these features; the scaling operation divided by $\sqrt{d_k}$ assures gradient stability, and $C$ denotes the relative position encoding that can be learned in $h_n$. V represents the set of values. SoftMax is applied to the scaled dot product to convert the similarity scores into a set of weights that sum to one.

The formula for the splicing fusion of the multi-headed self-attentive mechanism is as follows:

$$\mathrm{MH}(Q, K, V) = \mathrm{Cat}(h_1, \ldots, h_n)W^0 \tag{3}$$

where $W^0$ denotes the learnable weight matrix, $h_1, \ldots, h_n$ denotes the result of the computation from the first to the $n$th attention head. Cat concatenates the outputs of all the attention heads along the last dimension.

In order to realize the feature information transfer between different 3D Windows, the 3D Sliding Window Multi-head Self-Attention layer (3D SW-MSA) is utilized to perform sliding operations on the feature map $P^l$ after MLP and residual connection. In this process, each 3D window in the feature map $P^l$ is slid across by moving from the upper left corner to the lower right corner $N/2 \times M/4 \times M/4$. For each sliding position, the self-attention mechanism is recalculated for the new position of the 3D window. This sliding operation is then reversed, allowing the feature map's relative positions to be restored, resulting in the updated feature map $\tilde{P}^{l+1}$. Finally, the final feature map $P^{l+1}$ is output through LN and residual connection.

To guarantee the completeness of the features and extract them at various scales, the feature map $P^{l+1}$ after Spatial Swin Transformer Block is input to the Patch Merging layer for downsampling operation, and the feature map $P^{l+1}$ is segmented into 3D blocks with an interval of 2, and then, the channel splicing operation is used to halve the scale of the feature map and is input to the normalization layer for normalization processing, yielding a feature map with the length, width, and height halved and the number of channels doubled $\tilde{F}^{'}$. The 3D feature extraction module is divided into four stages, as shown in Figure 1. The first and second stages include Patch Merging and two Spatial Swin Transformer modules, and the third stage includes Patch Merging and six Spatial Swin Transformer modules. There are only two Spatial Swin Transformer Blocks in the fourth stage. Finally, the Spatial Swin Transformer Block in the fourth stage outputs the finally extracted three-dimensional feature map $\tilde{F}^{'}_{3D}$.

### *2.2. Feature Projection and 2D Feature Extraction Module*

In the projection module, the calculation formula of coordinate transformation is used to convert the original point cloud $F$ into a two-dimensional feature map $F^{'}$. The coordinate system of the point cloud is the global coordinate system, denoted as $(X_w, Y_w, Z_w)$. The first step is to transform it into the camera coordinate system, represented as $(X_c, Y_c, Z_c)$. The equation is as follows:

$$(X_c, Y_c, Z_c) = R(X_w, Y_w, Z_w) + T \tag{4}$$

where $T$ and $R$ stand for the offset vector and rotation matrix, respectively. Then, the camera coordinate system is converted to the image coordinate system $(X, Y)$ to generate a 2D feature map $P^{'}$, in the equation shown as follows:

$$(X, Y, 1) = U(X_c, Y_c, Z_c) \tag{5}$$

where $U$ denotes the camera internal reference matrix. In order to enhance the validity of the 2D feature map, the weight of every point is finally calculated using the sigmoid function; the formula is as follows:

$$\tilde{F}^{'} = f_{sigmoid}(F^{'}) \tag{6}$$

where $F^{'}$ denote the 2D feature map, $\tilde{F}^{'}$ denoted the weighted 2D feature map, and $f_{sigmoid}$ denotes the weighted calculation of the 2D features.

To enhance the completeness of extracting 2D feature information, the projected image with a size of $H \times W \times 3$ is input into the patch segmentation layer, where adjacent pixels in a $4 \times 4$ neighborhood are grouped into individual patches. These patches are

then flattened along the channel dimension, resulting in a dimension of $H/4 \times W/4 \times 48$. Subsequently, a linear embedding layer carries out linear transformations on the channel data for every pixel, altering the image dimension to $H/4 \times W/4 \times C$. Additionally, every sample undergoes normalization across the feature dimension. The Swin Transformer employs a W-MSA module to derive 2D features across various scales, improving the completeness of the 2D features. Patch Merging uses a patch splicing layer between each stage to downsample the image, divide adjacent pixels with an interval of 2 into patches, and then perform channel splicing operations, so that the feature map scale is halved. The 2D feature extraction module is also divided into four stages, as shown in Figure 1. The first and second stages include Patch Merging and two Swin Transformer modules, and the third stage includes Patch Merging and six Swin Transformer modules; only two Spatial Swin Transformer Blocks are included in the fourth stage. Finally, the Spatial Swin Transformer Block in the fourth stage outputs the final extracted two-dimensional feature map $\tilde{F}'_{2D}$.

### *2.3. The 2D/3D Feature Fusion Module*

To further optimize the extracted point cloud feature information and ensure the completeness of the features, we constructed a 2D/3D feature fusion module. This module performs feature fusion between the 3D feature $\tilde{F}'_{3D}$ and the 2D feature $\tilde{F}'_{2D}$ by employing point-wise addition and vector concatenation operations within the channel dimension. This process increases the feature information along each dimension of the feature map, resulting in the final fused 3D feature map $F'_{out}$, as shown in Figure 3.

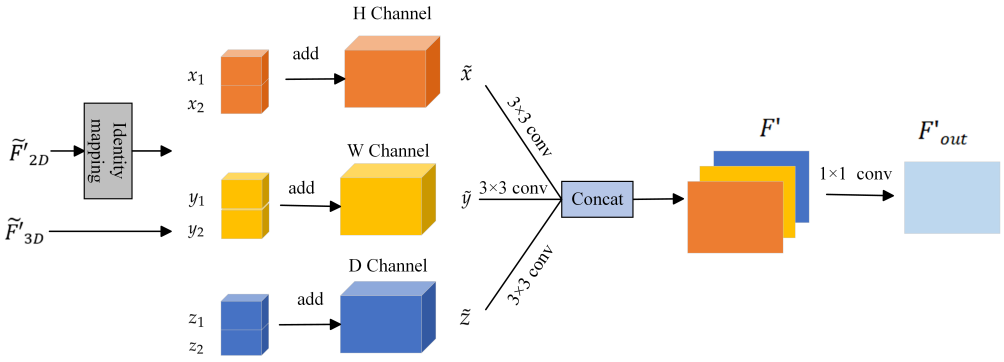



**Figure 3.** Feature fusion module.

To achieve feature fusion, firstly the 2D feature $\tilde{F}'_{2D}$ and 3D feature $\tilde{F}'_{3D}$ are input to the fusion module separately, the features of each point in the 3D feature map $\tilde{F}'_{3D}$ are represented as $(x1, y1, z1, r1, g1, b1)$, and the features of each point in the 2D feature map $\tilde{F}'_{2D}$ are represented as $(x2, y2, z2, r2, g2, b2)$. As the 2D feature map $\tilde{F}'_{2D}$ is missing feature information along the $z$-axis, the missing $z$-axis feature information $z2$ is zeroed out using the identity mapping operation. Subsequently, the feature information across distinct dimensions of the $x$, $y$, and $z$ axes are fed into $H$, $W$, and $D$ channels correspondingly, and add fusion operation is applied to the features within each channel, aiming to augment the feature information in every dimension, and the feature maps $\tilde{x}$, $\tilde{y}$, $\tilde{z}$ are obtained, respectively, after fusion. The equation is as follows:

$$\hat{x} = (x_1 + x_2) * k_x \tag{7}$$

$$\hat{y} = (y_1 + y_2) * k_y \tag{8}$$

$$\hat{z} = (z_1 + z_2) * k_z \tag{9}$$

where $x_1, y_1, z_1$ denotes feature information across various dimensions of the 3D feature map, $x_2, y_2, z_2$ denotes feature information across different dimensions of the 2D feature

map, $k_x, k_y, k_z$ denotes convolution kernel for various dimensions, and $*$ signifies the convolution operation.

Then, $3 \times 3$ depthwise separable convolution is applied to extract features within each channel, where filters are applied to each input channel individually. The number of convolutional filters is equal to the number of channels, ensuring a one-to-one correspondence between channels and filters. Consequently, each channel of the input layer undergoes convolution independently. Thus, a three-channel image generates three feature maps after the operation, with a total of 27 parameters $(3 \times 3 \times 3)$, as shown in Figure 4.

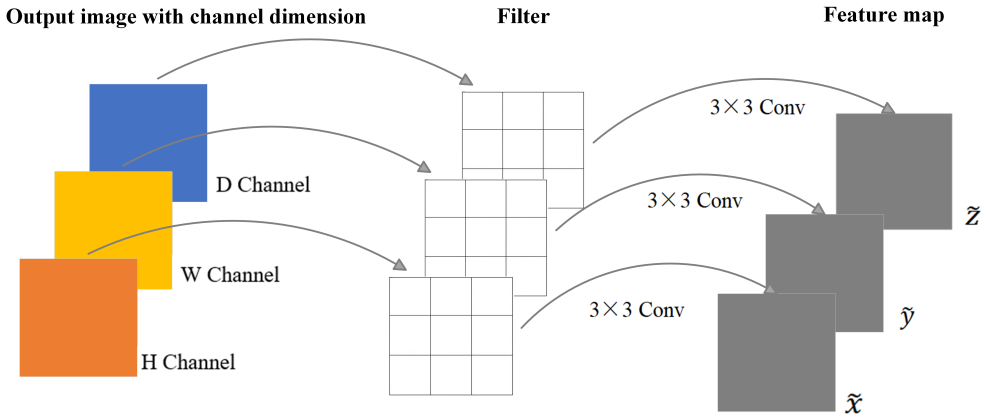



**Figure 4.** Depth-separable convolution.

The three generated feature maps are concatenated together to form a three-dimensional feature map with a channel size of 3. The calculation formula is as follows:

$$F' = \mathrm{Cat}(\hat{x}, \hat{y}, \hat{z}) \tag{10}$$

Ultimately, a $1 \times 1$ convolution is employed to alter the quantity of output channels and perform channel fusion on the feature maps obtained from the depthwise separable convolution. This process results in the fused 3D feature map $F'_{out}$, which combines the 2D and 3D features. It optimizes the extraction of point cloud feature information while ensuring the completeness of the features.

### *2.4. Detection Head*

The detection head module operates on the 3D feature map $F'_{out}$, derived from the feature fusion module's output. Initially, the module employs the Region Proposal Network (RPN) to categorize the input and calculate the positional offsets with respect to the initial point cloud coordinates, thereby yielding a set of preliminary bounding box proposals for potential objects. These proposals are subsequently channeled into the RoI Pooling layer, which aligns them with their corresponding locations on the feature map. The feature map is then segmented into uniform-sized regions, over which a maximum pooling operation is executed. This process ultimately leads to the creation of a set of detection boxes with predefined dimensions, crucial for the accurate identification and spatial positioning of objects within the scene.

### *2.5. Loss Function*

The loss function setting is typically composed of two components: classification loss and regression loss. The equation is as follows:

$$L_{rpn} = \frac{1}{N_{fg}} \sum_i L_{cls}(p_i^a, f_i^*) + I(f_i^* \geq 1) \sum_i L_{reg}(c_i^a, R_i^*) \tag{11}$$

where $L_{rpn}$ is the total loss for the Region Proposal Network (RPN), $N_{fg}$ denotes the count of foreground anchors, $L_{cls}$ represents the classification loss, $P_i^a$ and $c_i^a$ correspond to the outputs from the classification and regression branches, $f_i^*$ and $R_i^*$ denote the classification labels and regression targets, and $L_{reg}$ represents the regression loss. $I(f_i^* \geq 1)$ denotes the regression loss calculated using only foreground anchors. The classification loss is calculated with Focal loss [15], and the regression loss is computed using Huber Loss [16].

The second part is the losses of detection head; the equation is as follows:

$$L_h = \frac{1}{N_s} \sum_i L_{cls}(p_i^a, l_i^*) + I(IOU_i \geq \theta_{reg}) \sum_i L_{reg}(c_i^a, R_i^*) \tag{12}$$

where $L_h$ is the total loss for the detection head, $N_s$ signifies the quantity of suggestion frames selected for the training process; $I(IOU_i \geq \theta_{reg})$ indicates that only suggestion frames with IOU greater than or equal to $\theta_{reg}$ are involved in the regression loss calculation.

## 3. Experimental Section

To verify the efficacy of our method, extensive experiments were performed on the KITTI and nuScenes datasets, and we analyzed the results of the experiments in comparison with other method. The model was trained on Ubuntu 20.05 and NVIDIA GeForce 3090 GPU, using Adam as the optimizer, with a batch size of 4 and a learning rate of 0.01.

### *3.1. Dataset and Evaluation Metrics*

The KITTI dataset [17] stands as the predominant dataset in the realm of 3D object detection and segmentation. The dataset contains 7481 training samples, partitioned into training and test sets at an approximate 1:1 ratio, featuring 3712 samples for training and 3769 for testing. In this paper, the model is evaluated on the test set for the three classes of cars, pedestrians, and bicyclists trained. For each class, three difficulty levels were classified according to the object's size and the extent of its occlusion: easy, medium, and difficult. APs are calculated for cars, pedestrians and cyclists by means of 40 recall thresholds (R40).

The NuScenes dataset [18] consists of 1000 scenes containing images from six cameras. It is the latest popular benchmark for vision-based 3D object detection. As the nuScenes dataset is too large, the nuScences mini dataset with 10 scenes was chosen for this paper. The official evaluation metrics include mean accuracy (mAP), mean translation error (mATE), mean scale error (mASE), mean orientation error (mAOE), mean velocity error (mAVE), mean attribute error (mAAE), and NuScenes detection score (NDS). mAP is the average value of the detection accuracy. Half of the NDS is based on detection performance (mAP), while the other half is based on detection quality (ATE, ASE, AOE, AVE, AAE) based on position, size, orientation, attribute, and velocity measures of detection performance. The remaining metrics are used to calculate the accuracy of the results for the corresponding aspects (e.g., translation, scaling, orientation, velocity, and attributes). ↑ means higher values are better, ↓ means lower values are better.

### *3.2. Performance on the KITTI Dataset*

#### 3.2.1. Performance Measures

To evaluate the performance of our proposed method, we executed a set of experiments on the KITTI dataset, aiming to benchmark the performance of our approach against leading algorithms in the field. The experimental comparison results are shown in Table 1. The comparison algorithms include PointPillar [19], SECOND [7], SA-SSD [20], PV-RCNN [21], and Voxel-RCNN [22].

Unlike Voxel-RCNN and SECOND, which rely on voxel-based methods and utilize 3D Convolutional Neural Networks (3D CNN) for feature extraction, potentially leading to spatial information loss during voxelization, especially at lower resolutions, PointPillar employs a pillar-based structure and uses 2D CNN, which limits its ability to capture 3D spatial information. Our method employs a multi-scale feature extraction module based

on the 3D Swin Transformer, which effectively extracts spatial features at different scales through 3D Patch Partition and self-attention mechanisms, thus preserving more 3D spatial information and enhancing the accuracy and completeness of feature extraction. Furthermore, the proposed 2D/3D feature fusion module integrates 2D and 3D features through point-wise addition and vector concatenation, thereby enhancing the expressiveness and completeness of the features. Experimental results indicate that compared to current SOTA methods, our method improves average precision in car detection by 0.82%, 0.86%, and 1.34% for easy, moderate, and hard difficulty levels, respectively. In pedestrian detection, the average precision improves by 5.07%, 4.78%, and 4.36% across the same difficulty levels. For cyclist detection, the average precision increases by 2.55%, 3%, and 3.63%. Since pedestrians and cyclists are typically smaller in size and often appear in complex environments, these characteristics make pedestrian detection more challenging than car detection. Our proposed multi-scale feature extraction module and 2D/3D feature fusion module more effectively capture and integrate these complex spatial features, enhancing the accuracy of pedestrian and cyclist detection.

**Table 1.** Comparative analysis of various methods within the KITTI dataset.

| **Method** | **PointPillar** | **SECOND** | **SA-SSD** | **PV-RCNN** | **Voxel-RCNN** | **Proposed** |
|---|---|---|---|---|---|---|
| Car 3D $AP_{R40}$ | | | | | | |
| Easy | 87.75 | 90.97 | 92.32 | 92.57 | 92.38 | 93.20 |
| Mod | 78.39 | 79.94 | 84.30 | 84.83 | 85.29 | 86.15 |
| Hard | 75.18 | 77.09 | 81.36 | 81.36 | 82.86 | 84.20 |
| Car.mAP | 80.44 | 82.67 | 85.99 | 86.70 | 86.84 | 87.85 |
| Pedestrian 3D $AP_{R40}$ | | | | | | |
| Easy | 57.30 | 58.01 | – | 64.26 | – | 69.33 |
| Mod | 51.41 | 51.88 | – | 56.67 | – | 61.45 |
| Hard | 46.87 | 47.05 | – | 51.91 | – | 56.07 |
| Ped.mAP | 51.86 | 52.31 | – | 57.61 | – | 62.28 |
| Cyclist 3D $AP_{R40}$ | | | | | | |
| Easy | 81.57 | 78.50 | – | 88.88 | – | 91.43 |
| Mod | 62.94 | 56.74 | – | 71.95 | – | 74.95 |
| Hard | 58.98 | 52.83 | – | 66.78 | – | 70.41 |
| Cyc.mAP | 67.83 | 62.69 | – | 75.87 | – | 78.93 |

#### 3.2.2. Visualization Analysis

We present the illustration of the 3D target detection algorithm applied to the KITTI dataset in Figure 5, where three scene sets are processed, each consisting of four image sets: RGB images, Ground-Truth, Voxel-Rcnn, and our proposed method. The point cloud visualization examples in Figure 5a show that the detection networks introduced in this study effectively capture the information related to the car category and improve the accuracy of car detection. In the scene of Figure 5b, the Voxel-Rcnn network shows a large number of false detection results, as shown by the rectangular box in Figure 5b, which identifies clutter into categories such as car pedestrians, while the detection network in this paper has accurate false detection results. In the scene of Figure 5c, our method can detect the car target well and still correctly locate the car target under the occlusion problem, while the Voxel-Rcnn network locates the target position inaccurately. The above visualization results can visually show the effectiveness of our method.

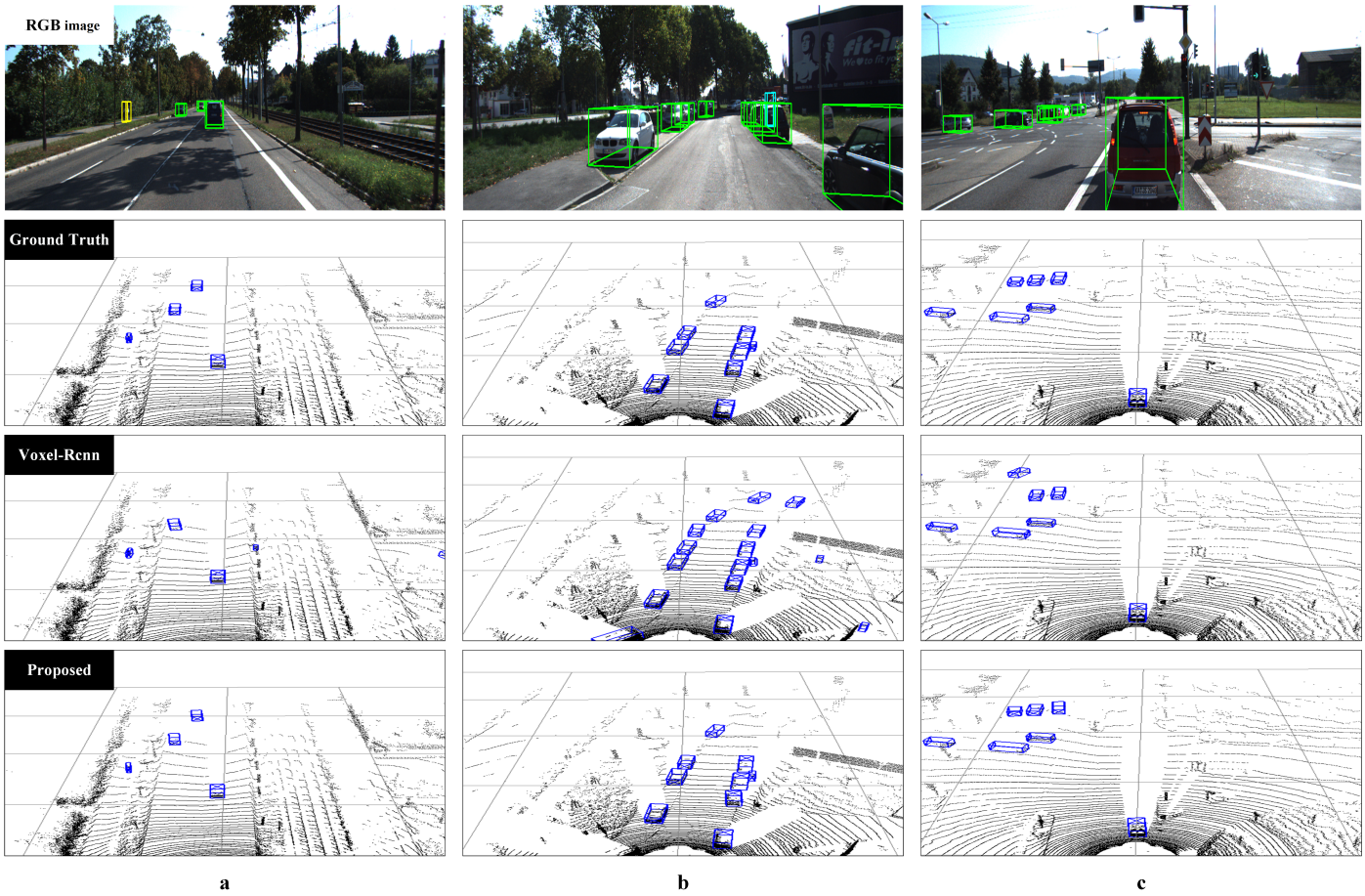



**Figure 5.** Comparison of visualisation results. The three figures (**a**–**c**) are the visualization of different scenarios, from which we can see that our proposed method has the best detection performance.

### *3.3. Performance on the NuScenes Dataset*

To additionally validate the practicality of our proposed method, experimental evaluations were carried out on the nuScenes (mini) dataset. The results of the experimental comparison are shown in Table 2. The comparison algorithms include PointPillar-MultiHead, SECOND-MultiHead, CenterPoint-PointPillar [23], CenterPoint (voxel_size = 0.1), and CenterPoint (voxel_size = 0.075). Our proposed method shows a decrease in five evaluation metrics (mATE, mASE, mAOE, mAVE, and mAAE), indicating better performance in terms of translation, scale, orientation, velocity, and attribute errors. Our proposed method achieves an increase in mAP and NDS, indicating improved detection accuracy and overall performance. Specifically, the proposed method outperforms other algorithms by achieving higher mAP and NDS scores while maintaining lower error rates in the other metrics.

**Table 2.** Comparative analysis of various methods within the NuScenes dataset. Bolded are the scores of our methods. ↑ means higher values are better, ↓ means lower values are better.

| Method | mATE ↓ | mASE ↓ | mAOE ↓ | mAVE ↓ | mAAE ↓ | mAP ↑ | NDST ↑ |
|---|---|---|---|---|---|---|---|
| PP-MH | 0.6339 | 0.5367 | 1.3782 | 1.3903 | 0.5340 | 0.1075 | 0.1833 |
| S-MH | 0.7366 | 0.5380 | 1.0812 | 1.5363 | 0.5401 | 0.1001 | 0.1686 |
| CP-PP | 0.6130 | 0.5306 | 1.1604 | 0.9568 | 0.3921 | 0.1467 | 0.2241 |
| CP (0.1) | 0.6105 | 0.5215 | 1.1065 | 1.1405 | 0.3390 | 0.1487 | 0.2272 |
| CP (0.075) | 0.4943 | **0.5076** | 1.0510 | 0.5977 | 0.3237 | 0.2546 | 0.3350 |
| **Proposed** | **0.5013** | 0.5081 | **1.0322** | **0.5939** | **0.3129** | **0.2637** | **0.3425** |

### *3.4. Ablation Experiments*

The ablation studies were all performed on the KITTI dataset, combining the experiments of the multi-scale features and feature fusion modules, using the performance of 3D $AP_{R11}$ on a moderately difficult car for evaluation, as shown in Table 3. The experiments were organized into five separate categories, designated (a), (b), (c), (d), and (e), for training. Among them, method (a) is the base model, which only linearly transforms the point cloud data, does not use multi-scale extracted features, and only 2D features are used to generate the detection frame candidate area. Methods (b), (c), and (d) build upon method (a) by sequentially introducing an additional stage, wherein the resolution of the feature map is reduced by half and the number of channels is doubled with each subsequent stage, use multi-scale extraction of features, have a larger perceptual field, and have a better ability to extract targets of different sizes; methods (b), (c), and (d) improve the 3D $AP_{R11}$ by 2.13%, 3.84%, and 4.8% in turn, 3.84% and 4.04%. Method (e), which adds the feature fusion module to method (d), is our proposed complete network architecture and enhances the accuracy of the detection frame candidate area by fusing the data of BEV features and multi-scale features after extracting features at multiple scales.

**Table 3.** Ablation studies on the KITTI dataset.

| Item | Setting | (a) | (b) | (c) | (d) | (e) |
|---|---|---|---|---|---|---|
| Multi-scale features | Stage-1 | ✓ | ✓ | ✓ | ✓ | ✓ |
| | Stage-2 | | ✓ | ✓ | ✓ | ✓ |
| | Stage-3 | | | ✓ | ✓ | ✓ |
| | Stage-4 | | | | ✓ | ✓ |
| Features fusion | Bev | ✓ | ✓ | ✓ | ✓ | ✓ |
| | BeV + Multi-scale features | | | | | ✓ |
| Result (%) | Moderate:AP3D | 75.26 | 77.39 | 81.23 | 85.27 | 86.93 |

## 4. Conclusions

We introduce a novel multi-scale feature fusion approach for point cloud object detection that leverages both the original point cloud data and its projection. Firstly, a 3D feature extraction module that employs the 3D Swin Transformer to distill features across various scales, enabling the network to adapt to inputs of arbitrary scales. This allows the network to capture the feature details of point clouds across multiple scales by having different receptive fields. Secondly, the point clouds are transformed into 2D images using projection, and 2D features are extracted using Swin Transformer. Subsequently, a fusion module for 2D/3D features is constructed, integrating the 2D and 3D features at the channel level through element-wise addition and vector concatenation, enhancing the completeness of feature information. Ultimately, the integrated feature maps are fed into the detection head, facilitating effective object detection. The experimental results demonstrate that our method achieves high accuracy and robustness in detecting objects of different sizes and shapes. Furthermore, the experimental results on different datasets demonstrate the good generalization capabilities of our method. Specifically, the multi-scale feature fusion and 2D/3D feature fusion techniques enable our method to maintain high detection performance under various environments and conditions. Future work will include testing on more live sensor data to further verify its online complexity and real-time performance.

**Author Contributions:** Conceptualization, Z.Z. (Zhikang Zhang) and Z.Z. (Zhongjie Zhu); methodology, Z.Z. (Zhikang Zhang) and Y.B.; validation, Z.Z. (Zhikang Zhang) and M.W.; resources, Z.Z. (Zhongjie Zhu); data curation, Z.Z. (Zhikang Zhang); writing—original draft preparation, Z.Z. (Zhikang Zhang); writing—review and editing, Z.Z. (Zhikang Zhang), Z.Z. (Zhongjie Zhu), Y.B., Y.J. and M.W.; visualization, Z.Z. (Zhikang Zhang); supervision, Z.Z. (Zhongjie Zhu) and Y.J.; project

administration, Z.Z. (Zhongjie Zhu); funding acquisition, Z.Z. (Zhongjie Zhu). All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was supported in part by the National Natural Science Foundation of China (61671412); the Zhejiang Provincial Natural Science Foundation of China (LZ24F010004, LY21F010014); the Yongjiang Sci-Tech Innovation 2035 (2023-23); and the Ningbo Municipal Major Project of Science and Technology Innovation 2025 (2022Z076).

**Data Availability Statement:** Data is contained within the article.

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

## References

1. 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.
2. Zheng, Q.; Zhao, P.; Zhang, D.; Wang, H. MR-DCAE: Manifold regularization-based deep convolutional autoencoder for unauthorized broadcasting identification. *Int. J. Intell. Syst.* **2021**, *36*, 7204–7238. [CrossRef]
3. Li, B. 3d fully convolutional network for vehicle detection in point cloud. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 1513–1518.
4. Beltrán, J.; Guindel, C.; Moreno, F.M.; Cruzado, D.; Garcia, F.; De La Escalera, A. Birdnet: A 3d object detection framework from lidar information. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 3517–3523.
5. Yang, B.; Luo, W.; Urtasun, R. Pixor: Real-time 3d object detection from point clouds. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7652–7660.
6. Zhou, Y.; Tuzel, O. Voxelnet: End-to-end learning for point cloud based 3d object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4490–4499.
7. Yan, Y.; Mao, Y.; Li, B. Second: Sparsely embedded convolutional detection. *Sensors* **2018**, *18*, 3337. [CrossRef] [PubMed]
8. Kuang, H.; Wang, B.; An, J.; Zhang, M.; Zhang, Z. Voxel-FPN: Multi-scale voxel feature aggregation for 3D object detection from LIDAR point clouds. *Sensors* **2020**, *20*, 704. [CrossRef] [PubMed]
9. 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 June 2017; pp. 652–660.
10. 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*. [CrossRef]
11. 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.
12. He, L.; Valocchi, A.J.; Duarte, C.A. An adaptive global–local generalized FEM for multiscale advection–diffusion problems. *Comput. Methods Appl. Mech. Eng.* **2024**, *418*, 116548. [CrossRef]
13. Zheng, Q.; Zhao, P.; Wang, H.; Elhanashi, A.; Saponara, S. Fine-grained modulation classification using multi-scale radio transformer with dual-channel representation. *IEEE Commun. Lett.* **2022**, *26*, 1298–1302. [CrossRef]
14. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 10012–10022.
15. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal loss for dense object detection. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988.
16. Laina, I.; Rupprecht, C.; Belagiannis, V.; Tombari, F.; Navab, N. Deeper depth prediction with fully convolutional residual networks. In Proceedings of the 2016 Fourth International Conference on 3D Vision (3DV), Stanford, CA, USA, 25–28 October 2016; pp. 239–248.
17. 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.
18. 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.
19. 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.
20. He, C.; Zeng, H.; Huang, J.; Hua, X.S.; Zhang, L. Structure aware single-stage 3d object detection from point cloud. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2020; pp. 11873–11882.

21. 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.
22. 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, Virtual Event, 2–9 February 2021; Volume 35, pp. 1201–1209.
23. Yin, T.; Zhou, X.; Krahenbuhl, P. Center-based 3d object detection and tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 11784–11793.

**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.