electronics

MDPI

Article

Denoising and Reducing Inner Disorder in Point Clouds for Improved 3D Object Detection in Autonomous Driving

Weifan Xu, Jin Jin, Fenglei Xu *, Ze Li and Chongben Tao

School of Electronic and Information Engineering, Suzhou University of Science and Technology, No. 99 Xuefu Road, Suzhou 215009, China
* Correspondence: xufl@mail.usts.edu.cn

Abstract: In the field of autonomous driving, precise spatial positioning and 3D object detection have become increasingly critical due to advancements in LiDAR technology and its extensive applications. Traditional detection models for RGB images face challenges in handling the intrinsic disorder present in LiDAR point clouds. Although point clouds are typically perceived as irregular and disordered, an implicit order actually exists, owing to laser arrangement and sequential scanning. Therefore, we propose Frustumformer, a novel framework that leverages the inherent order of LiDAR point clouds, reducing disorder and enhancing representation. Our approach consists of a frustum-based method that relies on the results of a 2D image detector, a frustum patch embedding that exploits the new data representation format, and a single-stride transformer network for original resolution feature fusion. By incorporating these components, Frustumformer effectively exploits the intrinsic order of point clouds and models long-range dependencies to further improve performance. Ablation studies verify the efficacy of the single-stride transformer component and the overall model architecture. We conduct experiments on the KITTI dataset, and Frustumformer outperforms existing methods.

Keywords: 3D object detection; point cloud; transformer

Citation: Xu, W.; Jin, J.; Xu, F.; Li, Z.; Tao, C. Denoising and Reducing Inner Disorder in Point Clouds for Improved 3D Object Detection in Autonomous Driving. *Electronics* **2023**, *12*, 2364. https://doi.org/10.3390/electronics12112364

Academic Editors: Krzysztof Okarma and Stefanos Kollias

Received: 2 April 2023
Revised: 11 May 2023
Accepted: 22 May 2023
Published: 23 May 2023

1. Introduction

The growing demand for accurate spatial positioning has led to 3D object detection becoming a crucial task in the field of autonomous driving. The declining cost and increasing resolution of LiDAR technology have generated interest in various applications requiring precise environmental perception, such as autonomous driving and simultaneous localization and mapping. Many of these applications employ laser sensors mounted on mobile platforms to capture data and perceive the environment from a fixed viewpoint. However, due to LiDAR installation location limitations, only portions of the surfaces facing the laser sensors can be measured, complicating the utilization of point cloud data, particularly for 3D object detection. Estimating oriented 3D bounding boxes that fully enclose targets is an essential task in point-cloud applications, and 3D object detection is a particularly active area of research in this domain. In this paper, we propose a workflow primarily focused on utilizing point cloud data for 3D object detection, aiming to enhance model performance by embedding frustum partitions and fusing global context information using the transformer architecture [1].

Although substantial research has been conducted on 3D object detection [2,3], most approaches assume that point clouds are discrete, unordered, and sparse, making it difficult for traditional detection models designed for RGB images to directly operate on 3D object detection tasks. To address this challenge, researchers tried to project point clouds onto the image plane [4,5] through cross-sensor calibration or transforming 3D expressions to 2D by grid map generation [6,7]. While applying 2D convolutional operations to such transformed data is feasible, it often results in the loss of valuable 3D information due to the dimensional reduction. Over the last three years, there has been a development of networks

that extract features directly from 3D space [8]. To make sparse point clouds continuous, some methods convert point clouds to new 3D representations, such as VoxelNet [9], SECOND [10], and Depth Map [11]. Methods commonly employed to process LiDAR raw data directly include PointNet [12] and PointNet++ [13]. Nonetheless, all the techniques mentioned earlier presuppose a disorganized point cloud during data manipulation.

In fact, LiDAR point clouds exhibit inherent order, stemming from their operational mode, although they may still display sparse or discrete characteristics due to equipment limitations. For instance, the Velodyne-64E LiDAR system features a fixed laser head arrangement and constant spinning speed, enabling all lasers to naturally sample points in an orderly fashion. Similarly, 2D images, such as those provided by the KITTI dataset, possess inherent order due to the regular arrangement of pixels, with all corresponding points having neighbors and context information. Inspired by this observation, we introduce the novel end-to-end object detection framework Frustumformer with the following key contributions:

- To address the disorder in point clouds, we adopt a frustum-based approach. This method relies on the results generated by the 2D image detector, and due to the constraints imposed by this 2D detector, the disorganization within the point cloud is effectively mitigated.
- Frustumformer leverages the inherent order of LiDAR point clouds and the transformer architecture for modeling long-range dependencies, enhancing 3D object detection performance. Additionally, the transformer architecture is employed to facilitate information interaction at a distance and obtain an adequate perceptual field to complete the semantic information for the incompletely displayed detection target, addressing the problem of missing information about detection targets due to the presence of nearby objects that obscure them.
- In this work, we employ a single-stride transformer network throughout the architecture to maintain the original resolution of the network. With the assistance of the transformer network, we effectively address the issue of insufficient receptive fields in single-stride structures. Additionally, this approach aligns well with the frustum proposal's characteristic of having a limited number of point clouds, naturally circumventing expensive computations.

The remainder of this paper is organized as follows: Section 2 discusses related work; Section 3 presents the details of the proposed Frustumformer model; Section 4 provides experimental validation of the model's efficiency; and Section 5 offers conclusions and future observations.

2. Related Works

Three-dimensional object detection is a highly regarded research objective in the field of artificial intelligence [14,15] and plays a significant role in pattern recognition [16,17]. Current 3D object detection models are primarily divided into two categories: those that utilize image inputs and those that rely on LiDAR data. Within the LiDAR-based models, there are two further subdivisions: those that convert 3D point clouds to images/voxels, and those that operate directly on raw data. Extensive research has been conducted in each of these areas, utilizing traditional data-processing methods as well as machine learning [18], deep learning [19,20], and swarm intelligence-based approaches [21,22].

2.1. Object Detection Based on Images

Three-dimensional object detection based on images is much more challenging than two-dimensional object detection [23] due to the ambiguities arising from 2D-3D mapping. In [24], a standard CNN pipeline is employed after generating a set of candidate object proposals belonging to specific classes on the ground plane to achieve high-quality object detection. Guo et al. [25] proposed a multi-scale feature fusion technique for 3D object detection. Mousavian et al. [26] incorporated geometry information to improve 3D object detection and utilized a multi-bin architecture to accurately generate orientation regression.

Additionally, this model includes tight correspondence constraints to achieve accurate 3D translation. Kundu et al. [27] trained a deep CNN to handle all object instances within the 2D image while simultaneously predicting their corresponding shape and pose in the 3D space. Xu and Chen [28] proposed using monocular depth estimation to improve localization accuracy, allowing for the better prediction of object shapes and poses in 3D space. Additionally, Chabot et al. [29] introduced FQNet, which uses 2D cues to determine the 3D IoU between the 3D proposals and the object. This is achieved by projecting a large number of candidates in 3D space onto a 2D image and selecting the best one based on spatial overlap exploration.

2.2. Object Detection Based on LiDAR Data

2.2.1. Methods Based on Converting to Images/Voxels

Chen et al. [4] described MV3D, which adopts a two-fold approach to 3D object detection from point clouds. Firstly, it converts the point cloud data into two different views: a bird's-eye view and a front view. These views are then used to train a region proposal network (RPN) that generates positive proposals for further processing. Secondly, Faster-RCNN [30] is employed for 3D object detection using these two views from the point cloud as well as RGB input data to generate refined 3D bounding boxes. AVOD [5] builds upon MV3D by improving both the encoder and decoder components of the network while also replacing ROI pooling with crop and resize operations, resulting in better detection performance. AVOD differs from MV3D, as it integrates multi-view features starting from the RPN phase rather than just in the refinement stage. A study by Asvadi et al. [31] examined and assessed 3D-LIDAR modalities that incorporate both depth and reflectance map modalities for identifying vehicles.

Voxel-grid is a different approach for representing LiDAR data in 3D object detection. In DSS [32], the point clouds are converted into depth images and then encoded as 3D voxels using the truncated signed distance function (TSDF). This allows for the use of 3D CNNs to classify objects and estimate their bounding boxes. PIXOR [7] uses occupancy to encode each voxel grid and employs handcrafted representation methods. VoxelNet [9] utilizes stacked VFE (voxel feature encoding) layers for feature extraction, which generates machine-learned representation for each voxel. In terms of processing efficiency, SECOND [10] differs from VoxelNet by utilizing sparse convolution layers to parse compact representations. PointPillars [33] is an extension of SECOND that improves upon voxel representation by implementing pillars, which generate pseudo feature images, enabling the use of 2D convolutions. However, these methods focus primarily on larger objects and may have difficulty detecting smaller objects, such as pedestrians and cyclists. In contrast, the SST [34] model demonstrates superior performance in detecting small objects. As a single-stride model, it performs sparse operations on non-empty voxels and generates bounding boxes on sparse point clouds, effectively highlighting the proportion of small objects relative to the size of the input scene.

2.2.2. Methods Based on Directly Operating on Raw Data

The use of raw data for object detection was first introduced by PointNet and PointNet++ [13]. However, these methods primarily target classification and segmentation tasks. To extend the application of PointNet to 3D object detection, Frustum-PointNet [35] was developed. This method starts by conducting 2D detection for region proposals, and then utilizes interior points to estimate oriented boxes. By doing so, Frustum-PointNet enables accurate predictions of 3D objects within point clouds. Frustum-PointNet uses a simplified version of PointNet known as T-Net, as its main feature extraction module. To enhance the precision of object detection, other studies, such as IPOD [36] and PointRCNN [19], have proposed alternative proposal generation methods. IPOD generates target proposals based on every point in the object and utilizes PointNet++ as the backbone network for feature extraction. The proposal features are then generated by fusing these two components before predicting the box. PointRCNN utilizes the complete point cloud to generate proposals and

incorporates proposal location information by using the segmentation score of the central point for classification. F-ConvNet [37] is designed to leverage the geometric structure of a frustum. By utilizing a sequence of frustum slices along a truncated voxel axis and integrating them with FCN, the model is capable of effectively handling disordered point cloud data for 3D object detection.

In this paper, we focus on 3D object detection and incorporate some feature extraction operations from PointNet++ [13]. The proposed method reorganizes point clouds to construct contextual information and operates on raw point clouds directly.

3. Frustumformer

In this section, we present Frustumformer, an end-to-end object detection framework utilizing the Transformer architecture for modeling long-range dependency as shown in Figure 1. First, we extract a 3D frustum point cloud from the 2D image detection results and partition it into frustum patches. We then employ PointNet for features extraction, followed by a single-stride Frustum Transformer network to fuse features across frustum patches. Building on the multi-resolution sliding frustums approach of [37], we apply this procedure to different resolution branches and aggregate the results. Finally, the output is passed through another frustum transformer network and a detection head for box classification and location prediction. This approach enables Frustumformer to effectively process 3D data and improve object detection performance.

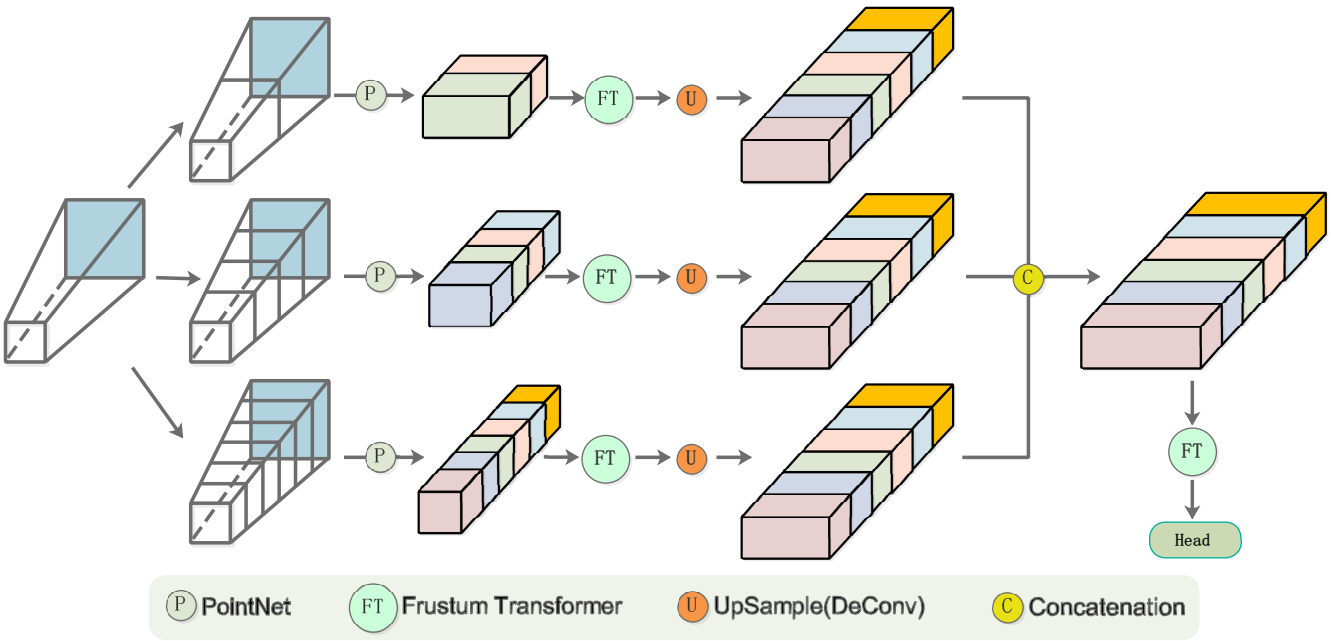

Figure 1. The Frustumformer framework: an end-to-end object detection system utilizing the Transformer architecture to model long-range dependencies in 3D frustum point clouds extracted from 2D image detection results. It incorporates PointNet feature extraction, single-stride frustum transformer networks for feature fusion, and multi-resolution sliding frustums for improved box classification and location prediction.

3.1. Frustum Patch Embedding

Employing an existing 2D object detector, we identify 2D region proposals in RGB images. By combining these proposals with the corresponding depth information, we obtain frustum proposals [35]. By sliding a pair of parallel planes perpendicular to the frustum axis with equal strides, we generate a series of frustum patches, which may overlap as illustrated in Figure 2. The stride for sliding along the frustum axis is set to s, and the height of each frustum patch is set to u. In our experimental section, we set $u = 2s$. For each 2D object detection region proposal, we generate a series of frustum patches. By employing

different stride lengths s on the proposed frustum, we create frustum patches with varying resolutions, forming multiple resolution branches that will be aggregated before being input into the detection head.

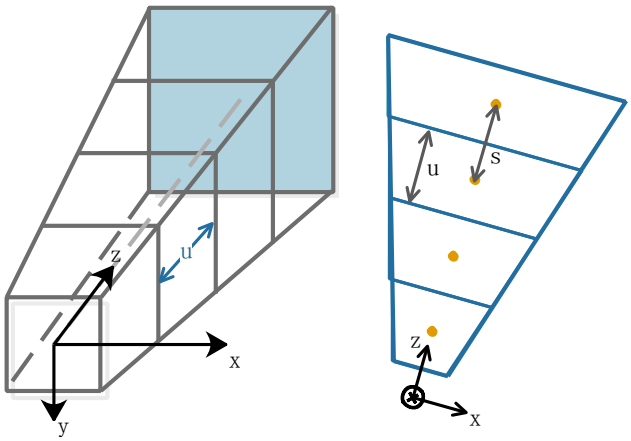

Figure 2. Frustum patches are generated by combining 2D region proposals from RGB images with depth information. Frustum patches are formed by sliding parallel planes along the frustum axis using a stride of s and a patch height of u. The z-direction represents the central axis of the slices, while the x-direction is parallel to the ground plane and perpendicular to the z-axis. The figure illustrates the case when $u = s$, allowing for overlapping between frustum patches.

Within a resolution branch, we use PointNet to process the raw point cloud in each frustum patch, extracting features to generate the frustum patch's feature encoding. Through the final max-pooling layer in the PointNet module, we obtain a single frustum patch feature of length d. Assuming we extract L frustum patches, the original frustum proposal will have a two-dimensional feature of length L, with dimensions $L \times d$. Let the feature vector of the frustum proposal be represented as $\{f_i\}_{i=1}^{L}, f_i \in R^d$. In subsequent processes, we further refine the frustum patch encoding using the frustum transformer and deconvolution techniques. The following section provides a detailed interpretation of the frustum transformer.

3.2. Single-Stride Frustum Transformer

Initially, the transformer architecture was primarily used in the natural language processing domain, capturing long-range dependencies in sequences through self-attention mechanisms. In recent years, it has also demonstrated exceptional capability in the computer vision field [38]. Inspired by [39], we design the frustum transformer for feature fusion between frustum patches. It plays a crucial role in the fusion of cross-frustum-patch features within each resolution branch and the final multi-resolution branch aggregation.

Compared to pixel-dense and regularly arranged 2D images, LiDAR point clouds exhibit variable sparsity: points close to the LiDAR sensor are densely packed, while those further away are much sparser. However, most existing works do not specifically consider the sparse point distribution of outdoor LiDAR point clouds, leading to inferior results for sparse distant points. One possible reason is that the model's receptive field cannot simultaneously cover near and far point clouds, as most works still expand their receptive field around points or voxels through downsampling. We propose that by radially expanding the receptive field and feeding the features of frustum patches at varying distances into the frustum transformer, we can leverage the self-attention mechanism to capture global context information, thus addressing the issue of covering both near and far point clouds effectively. In the experimental section, we demonstrate that Frustumformer, relying on the fusion of near and far information, performs well on partially occluded distant objects.

As illustrated in Figure 3, the frustum transformer block primarily consists of position encoding, normalization, multi-head self-attention (**MSA**) and feed-forward neural network (**FFN**) components. The following formula depicts this process, where **PE**(·) stands for the positional encoding function, **MSA**(·) denotes the multi-head self-attention, **NORM**(·) represents layer normalization, and **FFN** (·) means the feed-forward neural network:

$$\begin{cases} f' = \mathbf{MSA}(\mathbf{NORM}((\mathbf{PE}(f) + f)) + (\mathbf{PE}(f) + f) \\ \hat{f} = \mathbf{FFN}(\mathbf{NORM}(f')) + f' \end{cases} \tag{1}$$

We directly input the length L frustum patch feature sequence into the frustum transformer network. Given that different frustum patches represent different positions within the point cloud, and object position is critical for object detection tasks, we employ absolute position encoding for the input frustum patches sequence in the first frustum transformer block. We utilize standard learnable 1D position embeddings, which are added to the frustum patch features to preserve positional information.

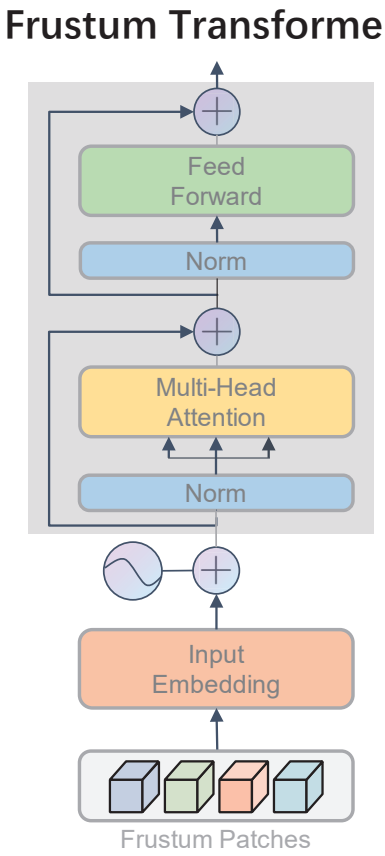

Figure 3. The frustum transformer block, designed for feature fusion between frustum patches or between multi-resolution branches, consisting of position encoding, normalization, multi-head self-attention, and feed- forward neural network components.

Within a single resolution branch, we employ the frustum transformer to achieve global context information fusion across frustum patches. Subsequently, we use deconvolution to upsample the frustum patch features of different resolution branches to the highest resolution, facilitating the concatenation of multi-resolution features. Then, the frustum transformer is used again to fuse multi-resolution features, with the final fused features being input into the detection head for classification and bounding box localization.

Furthermore, previous methods often employ downsampling operations to both expand the receptive field and reduce computational complexity. However, Frustumformer effectively reduces unnecessary computations and significantly narrows the point cloud

quantity by introducing frustum proposals, making it feasible to avoid downsampling altogether. Additionally, the self-attention mechanism within the frustum transformer inherently aids in capturing global context information without considering utilizing downsampling to expand the receptive field. Using downsampling would sacrifice the performance of high-resolution branches. Considering these factors, we propose the use of a single-stride frustum transformer, without any downsampling operations. In the experimental section, we demonstrate the superiority of using only a single-stride frustum transformer compared to the combined use of frustum transformer and downsampling in terms of performance.

3.3. Detection Header and Training of Frustumformer

The detection head follows the design employed in Frustum ConvNet, consisting of two parallel convolutional layers, which serve as the classification branch and regression branch, respectively. The classification branch adopts focal loss to address the imbalance between foreground and background samples. The regression branch includes the following components: the center regression loss is based on the Euclidean distance, while the size and angle offsets utilize smooth L1 regression loss, and corner loss is applied to regularize all bounding box regression parameters. The entire Frustumformer is trained using a total of three loss functions. It is worth noting that, as Frustumformer employs the transformer architecture, it has a higher model capacity and larger parameters, while optimization issues gradually emerge. Unlike ViT, which is trained on the large-scale ImageNet dataset [40], the KITTI dataset is considerably smaller in comparison. At this point, regularization is beneficial for optimizing the Frustumformer model. We find that regularization techniques effective in ViT, such as dropout, droppath, and high-value weight decay, all have a certain impact on improving the optimization of the Frustumformer model.

4. Experiments

4.1. Dataset

We implement the proposed model using Pytorch 1.10 on a graphics workstation with a Nvidia RTX 3090 GPU, primarily evaluating it on the KITTI-OBJECT dataset. This dataset contains 7481 training scenes and 7518 test scenes, with over 93,000 depth maps and corresponding raw LiDAR scans and RGB images. The dataset includes 2D, 3D, and bird's-eye views, and a benchmark of three types of view angles with a total of 80,256 labeled objects for performing target detection tasks. We focus our assessment on three primary objects: cars, pedestrians, and cyclists. These are evaluated across three difficulty levels: easy, moderate, and hard. The difficulty levels are determined based on the size of the target and the level of truncation. In line with the KITTI dataset guidelines, the difficulty levels (easy, moderate, and hard) are determined based on the target size, occlusion, and truncation. The definitions for each difficulty level are as follows: Easy—minimum bounding box height of 40 pixels, fully visible objects (no occlusion), and maximum truncation of 15%. Moderate—minimum bounding box height of 25 pixels, partly occluded objects, and maximum truncation of 30%. Hard—minimum bounding box height of 25 pixels, objects that are difficult to see (higher occlusion levels), and maximum truncation of 50%. Following the method MV3D [4], we use the same split on the original training set for supervised data. The new training and validation sets contain 3712 and 3769 driving scenes, respectively. An ablation study is conducted on the new data split, while the final result comparison with other existing models is based on the KITTI validation set. The IoU thresholds are set at 0.7, 0.5 and 0.5 for car, pedestrian, and cyclist, respectively.

4.2. Implementation Details

We train two separate networks for car and pedestrian/cyclist categories on the KITTI dataset due to their distinct differences. During the network training process, we utilize the RRC model [41] and MSCNN model [42] to generate initial 2D bounding boxes for the car and pedestrian/cyclist categories, respectively. To augment the data, we scale the size of

the 2D bounding box, implement random shifts and flips, and employ techniques similar to the Frustum PointNet method to prevent overfitting. We use a fixed number of 1024 points to normalize the input points. Positive and negative training samples are generated by shrinking the ground-truth boxes by 0.5. Anchor boxes with centers inside the shrunken ground-truth boxes are considered the foreground, while anchor boxes with centers locate between the shrunken and ground-truth boxes are ignored. The remaining anchor boxes are tagged as the background.

We employ two frustum transformer blocks within a single resolution branch and utilize an additional two frustum transformer blocks during the stage of multi-resolution branch fusion. We train the Frustumformer with a batch size of 96 on one GPU (Nvidia RTX 3090 24 GB) and use the AdamW optimizer with a weight decay of 0.05. The initial learning rate is set to 0.0005, followed by cosine annealing decay over a maximum of 50 epochs. A linear warm-up is applied during the first epoch. The valid range is set from 0 to 70 m. The parameters of Frustumformer for different object categories and different datasets are provided in Table 1. These parameters include frustum resolutions, strides, frustum patch feature depths, dimensions and unified feature dimensions after deconvolution. In the evaluation, NMS is used to reduce redundancy. The final score of a 3D bounding box equals the average of the 2D detection score and the predicted 3D score.

Table 1. Frustumformer parameters for various object categories and datasets.

Parameter	KITTI	
	Car	**Pedestrian/Cyclist**
Frustum resolutions (u)	[0.5, 1.0, 2.0, 4.0]	[0.2, 0.4, 0.8, 1.6]
Strides (s)	[0.25, 0.5, 1.0, 2.0]	[0.1, 0.2, 0.4, 0.8]
Frustum patch feature depths (d)	[128, 128, 256, 512]	[128, 128, 256, 512]
Dimensions (L)	[280, 140, 70, 35]	[700, 350, 175, 88]
Unified feature dimensions after deconvolution	140	350

4.3. Ablation Study

To assess the effectiveness of our proposed method, we evaluate Frustumformer on the KITTI-OBJECT dataset through an ablation study and efficiency comparison with state-of-the-art techniques. The ablation study is conducted on the car category, known for containing rich features, with a train/val split. Table 2 demonstrates the impact of the 2D proposal accuracy on the Frustumformer results without the refinement stage. We utilize the RRC model [41] to generate 2D region proposals using RGB images. "GT" in the table refers to utilizing the 2D detection ground truth as the initial region proposal.

Table 2. Impact of 2D proposal on average precision (AP) (%).

	2D Detection			**3D Detection**		
	Easy	**Mod.**	**Hard**	**Easy**	**Mod.**	**Hard**
RRC	96.32	95.18	88.94	85.73	76.10	67.49
GT	100	100	100	84.49	84.15	77.18

As per Table 2, using the RRC model for 2D detection on RGB images yields an accuracy of 76.10% for moderate targets. Replacing RGB 2D detection with ground truth images significantly increases the precision of the 2D detection results, leading to a dramatic increase in 3D detection outcomes. The precision of the corresponding 3D detection results is considerably influenced by the two different sources of 2D proposals. Generally, with the cooperation of RGB images, Frustumformer achieves higher performance, and better 2D region proposals significantly contribute to 3D detection.

To verify the effect of single stride and regularization on the proposed model, we conduct an ablation study on Frustumformer. The results in Table 3 confirm the efficacy of

each component in the proposed model. Both single-stride and regularization operations benefit the detection output. According to Table 3, the single-stride frustum transformer without down-sampling procedures causes a slight decrease in the bird's-eye view but clear improvements on the 3D bounding box precision. Regularization further increases model performance by approximately 0.5 percents points on each item.

Table 3. Effect of single stride and regularization on average precision (AP) (%).

	Bird's-Eye View			**3D Bounding Box**		
	Easy	**Mod.**	**Hard**	**Easy**	**Mod.**	**Hard**
w/o Single Stride w/o Regularization	87.86	86.06	77.54	80.65	72.70	64.92
w/o Single Stride	89.45	86.76	77.98	84.92	75.20	66.79
w/o Regularization	88.98	86.52	78.18	85.37	75.49	67.26
Frustumformer	89.35	87.27	78.34	85.73	76.09	67.49

4.4. Main Results

Table 4 shows the performance of the proposed method (Frustumformer) on the KITTI validation set, compared to other listed methods. According to Table 4, our proposed model has obvious advantages in car detection compared with other models on the KITTI dataset. It can be noted that the Frustumformer method is more sensitive to details because of the single-stride transformer network, and the detection accuracy of moderate and hard targets is significantly improved compared with other networks. Our method's high detection rate for partially occluded distant objects, as demonstrated by the fusion of near and far information in Frustumformer, is illustrated in Figure 4. However, if the target is too close and contains too many points, the redundant local details may affect the accuracy of the 3D detection.

Table 4. Comparison of 3D object detection and BEV detection on the KITTI val split set at IoU = 0.7 for cars. (%).

	3D Detection			**BEV Detection**		
	Easy	**Mod.**	**Hard**	**Easy**	**Mod.**	**Hard**
VeloFCN [43]	15.20	13.66	15.98	-	-	-
MV3D [4]	71.29	62.68	56.56	86.55	78.10	76.67
F-PointNet [35]	83.76	70.92	63.65	88.16	84.02	76.44
AVOD-FPN [5]	84.41	74.44	68.65	-	-	-
VoxelNet [9]	81.98	65.46	62.85	89.60	84.81	78.57
Frustumformer	85.73	76.09	67.49	89.35	87.27	78.34

5. Conclusions

We present a novel approach for 3D object detection in LiDAR point clouds by leveraging the inherent order of the data and employing a frustum-based method to reduce the inner disorder. Our proposed framework, Frustumformer, utilizes a single-stride transformer network to maintain the original resolution of the network, while effectively addressing the insufficient receptive fields issue inherent in single-stride structures. The proposed model's efficiency is confirmed by experiments on the KITTI dataset. With regard to future research, we will explore methods beyond the frustum-based approach to reduce dependency on high-quality 2D proposals. This will further lower data acquisition costs and enhance the performance of our model.

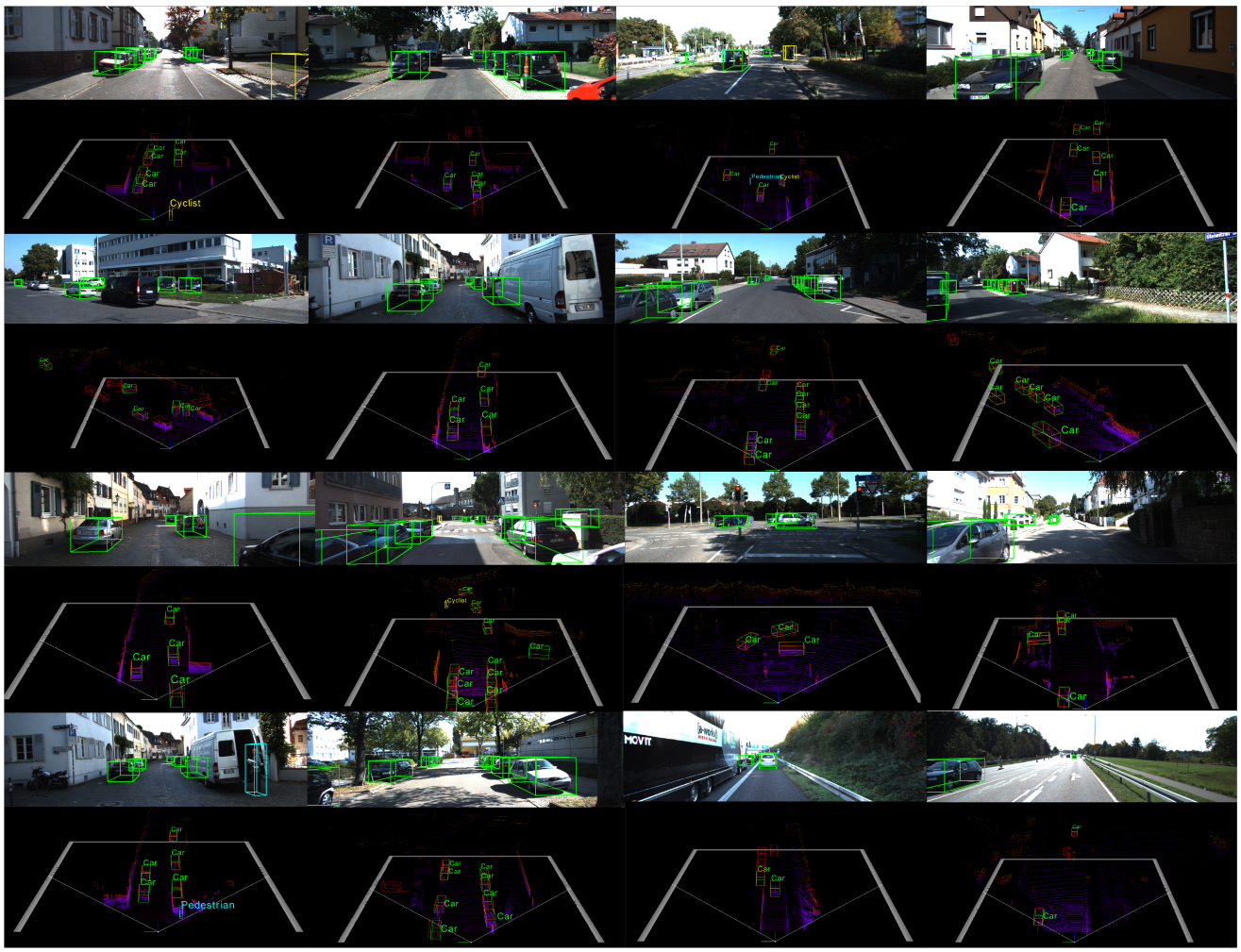

Figure 4. Qualitative results for the car category on KITTI dataset. We present 16 typical scenes, including images and LiDAR illustrations, to showcase the performance of our proposed method. Green bounding boxes represent ground truth, while red boxes indicate the proposed model predictions for car categories. Notably, our method demonstrates a high detection rate for objects, even in the presence of severe occlusions, by effectively fusing near and far information. This is particularly impressive, given that the KITTI dataset provides annotations for occluded objects but not for those with substantial occlusion. This highlights the robustness and generalizability of our approach in real-world scenarios and underscores its potential for improving the safety and efficiency of autonomous driving systems.

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

Funding: The research was funded by the Postgraduate Research and Practice Innovation Program of Jiangsu Province (SJCX21_1427) and General Program of Natural Science Research in Jiangsu Universities (21KJB520019).

Data Availability Statement: The KITTI dataset is available at https://www.cvlibs.net/datasets/kitti/eval_object.php (accessed on 1 April 2023).

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

References

1. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.U.; Polosukhin, I. Attention Is All You Need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., Eds.; Curran Associates, Inc.: New York, NY, USA, 2017; Volume 30.
2. Liang, W.; Xu, P.; Guo, L.; Bai, H.; Zhou, Y.; Chen, F. A survey of 3D object detection. *Multimed. Tools Appl.* **2021**, *80*, 29617–29641. [CrossRef]
3. Fernandes, D.; Silva, A.; Névoa, R.; Simoes, C.; Gonzalez, D.; Guevara, M.; Novais, P.; Monteiro, J.; Melo-Pinto, P. Point-cloud based 3D object detection and classification methods for self-driving applications: A survey and taxonomy. *Inf. Fusion* **2021**, *68*, 161–191. [CrossRef]
4. Chen, X.; Ma, H.; Wan, J.; Li, B.; Xia, T. Multi-View 3D Object Detection Network for Autonomous Driving. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1907–1915.
5. Ku, J.; Mozifian, M.; Lee, J.; Harakeh, A.; Waslander, S.L. Joint 3D Proposal Generation and Object Detection from View Aggregation. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 1–8.
6. Engelcke, M.; Rao, D.; Wang, D.Z.; Tong, C.H.; Posner, I. Vote3Deep: Fast object detection in 3D point clouds using efficient convolutional neural networks. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 1355–1361.
7. 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 (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 7652–7660.
8. Tao, C.; Fu, S.; Wang, C.; Luo, X.; Li, H.; Gao, Z.; Zhang, Z.; Zheng, S. F-PVNet: Frustum-Level 3-D Object Detection on Point–Voxel Feature Representation for Autonomous Driving. *IEEE Internet Things J.* **2023**, *10*, 8031–8045. [CrossRef]
9. 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 (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 4490–4499.
10. Yan, Y.; Mao, Y.; Li, B. SECOND: Sparsely Embedded Convolutional Detection. *Sensors* **2018**, *18*, 3337. [CrossRef] [PubMed]
11. Zhang, Y.; Lu, J.; Zhou, J. Objects are Different: Flexible Monocular 3D Object Detection. *arXiv* **2021**, arXiv:2104.02323. [CrossRef]
12. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 June 2017; pp. 652–660.
13. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Volume 30.
14. Tian, C.; Zheng, M.; Zuo, W.; Zhang, B.; Zhang, Y.; Zhang, D. Multi-stage image denoising with the wavelet transform. *Pattern Recognit.* **2023**, *134*, 109050. [CrossRef]
15. Tian, C.; Zhang, X.; Lin, J.C.W.; Zuo, W.; Zhang, Y.; Lin, C.W. Generative adversarial networks for image super-resolution: A survey. *arXiv* **2022**, arXiv:2204.13620.
16. Zhang, Q.; Xiao, J.; Tian, C.; Chun-Wei Lin, J.; Zhang, S. A robust deformed convolutional neural network (CNN) for image denoising. *CAAI Trans. Intell. Technol.* **2022**. [CrossRef]
17. Tian, C.; Xu, Y.; Zuo, W.; Lin, C.W.; Zhang, D. Asymmetric CNN for image superresolution. *IEEE Trans. Syst. Man Cybern. Syst.* **2021**, *52*, 3718–3730. [CrossRef]
18. Tombari, F.; Di Stefano, L. Object Recognition in 3D Scenes with Occlusions and Clutter by Hough Voting. In Proceedings of the 2010 Fourth Pacific-Rim Symposium on Image and Video Technology, Singapore, 14–17 November 2010; pp. 349–355. [CrossRef]
19. Shi, S.; Wang, X.; Li, H. PointRCNN: 3D Object Proposal Generation and Detection from Point Cloud. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 770–779.
20. Luo, X.; Zhou, F.; Tao, C.; Yang, A.; Zhang, P.; Chen, Y. Dynamic Multitarget Detection Algorithm of Voxel Point Cloud Fusion Based on PointRCNN. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 20707–20720. [CrossRef]
21. Bacanin, N.; Stoean, R.; Zivkovic, M.; Petrovic, A.; Rashid, T.A.; Bezdan, T. Performance of a Novel Chaotic Firefly Algorithm with Enhanced Exploration for Tackling Global Optimization Problems: Application for Dropout Regularization. *Mathematics* **2021**, *9*, 2705. [CrossRef]
22. Malakar, S.; Ghosh, M.; Bhowmik, S.; Sarkar, R.; Nasipuri, M. A GA based hierarchical feature selection approach for handwritten word recognition. *Neural Comput. Appl.* **2020**, *32*, 2533–2552. [CrossRef]
23. Chen, K.; Franko, K.; Sang, R. Structured Model Pruning of Convolutional Networks on Tensor Processing Units. *arXiv* **2021**, arXiv:2107.04191.
24. Chen, X.; Kundu, K.; Zhang, Z.; Ma, H.; Fidler, S.; Urtasun, R. Monocular 3D Object Detection for Autonomous Driving. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 2147–2156.
25. Guo, R.; Li, D.; Han, Y. Deep multi-scale and multi-modal fusion for 3D object detection. *Pattern Recognit. Lett.* **2021**, *151*, 236–242. [CrossRef]

26. Mousavian, A.; Anguelov, D.; Flynn, J.; Kosecka, J. 3D Bounding Box Estimation Using Deep Learning and Geometry. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 7074–7082.
27. Kundu, A.; Li, Y.; Rehg, J.M. 3D-RCNN: Instance-Level 3D Object Reconstruction via Render-and-Compare. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 3559–3568.
28. Xu, B.; Chen, Z. Multi-Level Fusion Based 3D Object Detection from Monocular Images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 2345–2353.
29. Chabot, F.; Chaouch, M.; Rabarisoa, J.; Teuliere, C.; Chateau, T. Deep MANTA: A Coarse-To-Fine Many-Task Network for Joint 2D and 3D Vehicle Analysis from Monocular Image. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2040–2049.
30. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
31. Asvadi, A.; Garrote, L.; Premebida, C.; Peixoto, P.; Nunes, U. Multimodal vehicle detection: Fusing 3D-LIDAR and color camera data. *Pattern Recognit. Lett.* **2018**, *115*, 20–29. [CrossRef]
32. Song, S.; Xiao, J. Deep Sliding Shapes for Amodal 3D Object Detection in RGB-D Images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 808–816.
33. 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 Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 12697–12705.
34. Fan, L.; Pang, Z.; Zhang, T.; Wang, Y.X.; Zhao, H.; Wang, F.; Wang, N.; Zhang, Z. Embracing single stride 3D object detector with sparse transformer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2022; pp. 8458–8468.
35. Qi, C.R.; Liu, W.; Wu, C.; Su, H.; Guibas, L.J. Frustum PointNets for 3D Object Detection from RGB-D Data. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 918–927.
36. Yang, Z.; Sun, Y.; Liu, S.; Shen, X.; Jia, J. IPOD: Intensive Point-based Object Detector for Point Cloud. *arXiv* **2018**, arXiv:1812.05276.
37. Wang, Z.; Jia, K. Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal 3d object detection. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 1742–1749.
38. 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, Nashville, TN, USA, 20–25 June 2021; pp. 10012–10022.
39. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.
40. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
41. Ren, J.; Chen, X.; Liu, J.; Sun, W.; Pang, J.; Yan, Q.; Tai, Y.W.; Xu, L. Accurate Single Stage Detector Using Recurrent Rolling Convolution. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 5420–5428.
42. Cai, Z.; Fan, Q.; Feris, R.; Vasconcelos, N. A Unified Multi-scale Deep Convolutional Neural Network for Fast Object Detection. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 10–11 October 2016; pp. 354–370.
43. 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. [CrossRef]