3DifFusionDet: Diffusion Model for 3D Object Detection with Robust LiDAR-Camera Fusion

Xinhao Xiang, Simon Dräger, Jiawei Zhang
IFM Lab, University of California, Davis
One Shields Avenue, Davis, CA, 95616
xinhao@ifmlab.org, sdraeger@ucdavis.edu, jiawei@ifmlab.org

Abstract

LiDAR-camera fusion techniques have demonstrated impressive performance in 3D object detection, mitigating each sensor's limitations while improving the detection system's robustness. Having shown appealing properties of flexibility, enabling dynamic and iterative evaluation, there is an increasing trend of applying the diffusion model in several computer vision tasks. While diffusion model has been shown well efficient in 2D Object Detection[10], its performance on 3D object detection tasks has not been explored yet. In this paper, we propose the 3DifFusionDet framework, which structures 3D object detection as a denoising diffusion process from noisy 3D boxes to target boxes. In this framework, ground truth boxes diffuse in a random distribution for training, and the model learns to reverse the noising process. During inference, the model gradually refines a set of boxes that were generated at random to the outcomes. Under the feature align strategy, the progressive refinement method could make a significant contribution to robust LiDAR-Camera fusion. The iterative refinement process could also demonstrate great adaptability by applying the framework to various detecting circumstances where varying levels of accuracy and speed are required. We explored extensively the best possible feature fusion strategies and finally proposed the fusion structure. Extensive experiments on KITTI, a benchmark for real-world traffic object identification, revealed that 3DifFusionDet is able to perform favorably in comparison to earlier, well-respected detectors.

1. Introduction

With the introduction of 3D sensors and a variety of 3D understanding applications, 3D recognition [71], object detection [25], and segmentation [50] have come under more scrutiny in research. 3D data is crucial for a wide range of applications, including navigation [21, 24], augmented reality [48], and robotics [23]. Among these tasks, 3D object detection is a critical issue and a key stage in many pipelines for 3D comprehension. Unlike traditional 2D object detection, 3D detection provides richer spatial information about objects, allowing for accurate depth perception and volumetric understanding. Those advantages are crucial in applications such as autonomous driving, where discerning the precise distance and orientation of surrounding vehicles is an essential matter of safety.

Most high-performance 3D object detectors nowadays utilize several sensors, including cameras and LiDAR, combining information to perform 3D object detection [12, 61, 74]. Combining the high-resolution visual cues from cameras with the depth information from LiDAR provides a more comprehensive scene representation, enhancing detection accuracy, especially in complex environments. Moreover, multi-modal fusion mitigates each sensor's limitations [6, 15]. It improves the detection system's robustness, making it less susceptible to errors or ambiguities from relying on a single sensor modality [43]. Owing to these significant advantages, multi-modal fusion is arguably the field's future trajectory.

There are two crucial elements worth investigating, the first being the 3D detecting head. Pre-defined anchor-box proposals [51, 59, 72] and learnable anchor-free queries [14, 57, 73] are examples of traditional methods. The first group of methods involves a pre-defined set of 3D bounding boxes, called anchors, of different shapes and sizes that slide across the spatial dimensions of the feature map. For the second group, instead of using pre-defined anchor boxes, these methods predict objects directly from feature points or use other mechanisms, such as sparse convolution windows [14, 65] or point bases [73].

Originating from the 2D object detection field, the query-based detection paradigm has recently received a lot of attention [19, 37, 64, 79], thanks to DETR's [9] proposal of learnable object queries to do away with the hand-designed components and build up an end-to-end detection pipeline. Several attempts in 3D object detection have been con-

ducted [6, 44]. These works manage a straightforward and efficient architecture but still depend on a predetermined set of learnable equations.

More recently, after observing tremendous success in several generation tasks [3, 5, 32, 66], diffusion models have been investigated in perception tasks like image segmentation [7, 11, 25, 26], text-video retrieval [33], human pose estimation [55]. DiffusionDet [10] proposes a novel framework that directly detects objects from a set of random boxes by using a diffusion model [30]. The underlying principle of their noise-to-box paradigm is analogous to the noise-to-image procedure observed in the denoising diffusion models [16, 30]. These models construct images by iteratively eliminating noise through a trained denoising mechanism, achieving good but improvable performance in 2D detection. The diffusion model has shown the appealing properties of flexibility, enabling dynamic numbers of boxes and iterative evaluation, which are all critical for the object detection tasks. However, its performance on 3D object detection tasks has not been well-explored yet.

In this paper, *we extend the usage of noise-to-image denoising pipelines to 3D object detection.* To exploit its potential benefits and performance as much as possible, we design our noise-to-3DBox paradigm under a LiDAR-camera fusion framework, which can provide a richer, more robust, and comprehensive detection paradigm. We structure 3D object detection as a denoising diffusion process [30] from noisy 3D bounding boxes to target boxes. In this framework, ground truth boxes diffuse in a Gaussian distribution for training. These noisy boxes extract Regions of Interest (RoI; [12, 29, 52, 59]) from the output feature map of the backbone encoder. These RoI features are forwarded to the detection decoder, trained to estimate the noise-free ground truth boxes by learning the reverse process. When the number of diffusion steps $D \in \mathbb{N}$ is large enough, e.g., $D = 1000$, the noisy boxes can be viewed as random variables sampled from the space of bounding boxes. During inference, several randomly generated bounding boxes are sampled for the learned reversing process to predict the 3D ground truth boxes.

Compared to using Lidar-only methods, we choose to adopt the fusion strategy due to its potential high performance by mitigating each sensor's limitations while improving the detection system's robustness. Therefore, the second element comes the camera-LiDAR fusion alignment strategies. Camera images and LiDAR cloud features are two inherently different features, where images are a dense data representation with most of the space in a point cloud being empty ("sparse"). Images contain rich texture and color information while lacking depth information. On the other hand, simply executing the RoIAlign operations [10, 29] on the fused features does not fully use the complementary information the two modalities provide. Drawing upon this, besides putting the point cloud RoIAlign under the encoded 3D features, we also create a second branch that performs an image RoIAlign under the encoded 2D features. As for the connection of these two feature branches, simply concatenating them will suffer from information cuts which lead to reduced performance. To this end, a multi-head cross attention mechanism [68] is introduced to align these features deeply. These aligned features are sent to the detection head in order to predict the ground truth boxes without noise.

We evaluate 3DifFusionDet on the KITTI 3D object detection benchmark [21]. With proper feature extraction and fusion backbones [14, 28, 61, 78], 3DifFusionDet achieves higher mean Average Precision (mAP), which outperforms several latest and popular methods. Different from all other SOTA 3D object detection methods, our 3DifFusionDet has the ability to perform multi-step inference once training has concluded. Besides that, our framework has the flexibility to reach different levels of detection accuracy and inference speed by changing the number of denoising sampling steps, both of which reveal broader potential usages.

Our contributions can be summarized as follows:

- We formulate 3D object detection as a generative denoising process and propose 3DifFusionDet, which to the best of our knowledge is the first study to apply the diffusion model to 3D object detection.
- We investigate the best camera-LiDAR fusion alignment strategies under the generative denoising process framework and propose 2-branch fusion align strategies to exploit the complementary information that the two modalities provide.
- Extensive experiments are conducted on the KITTI benchmark. 3DifFusionDet achieves competitive results compared with existing well-designed approaches, showing a promising future of diffusion models in 3D object detection tasks.

2. Related Work

2.1. 3D Object Detection with LiDAR-Camera Fusion

For 3D object detection, camera and LiDAR are two complementing sensor types. LiDAR sensors specialize in 3D localization and provide rich information about 3D structures, while cameras provide color information from which rich semantic features can be derived [43]. Many efforts have been made to accurately detect 3D objects by fusing data from cameras and LiDARs. State-of-the-art approaches [6, 42, 49, 74, 76] are primarily based on LiDAR-based 3D object detectors and strive to incorporate image information into various stages of a LiDAR detection pipeline since LiDAR-based detection methods perform significantly better than camera-based methods. Combining

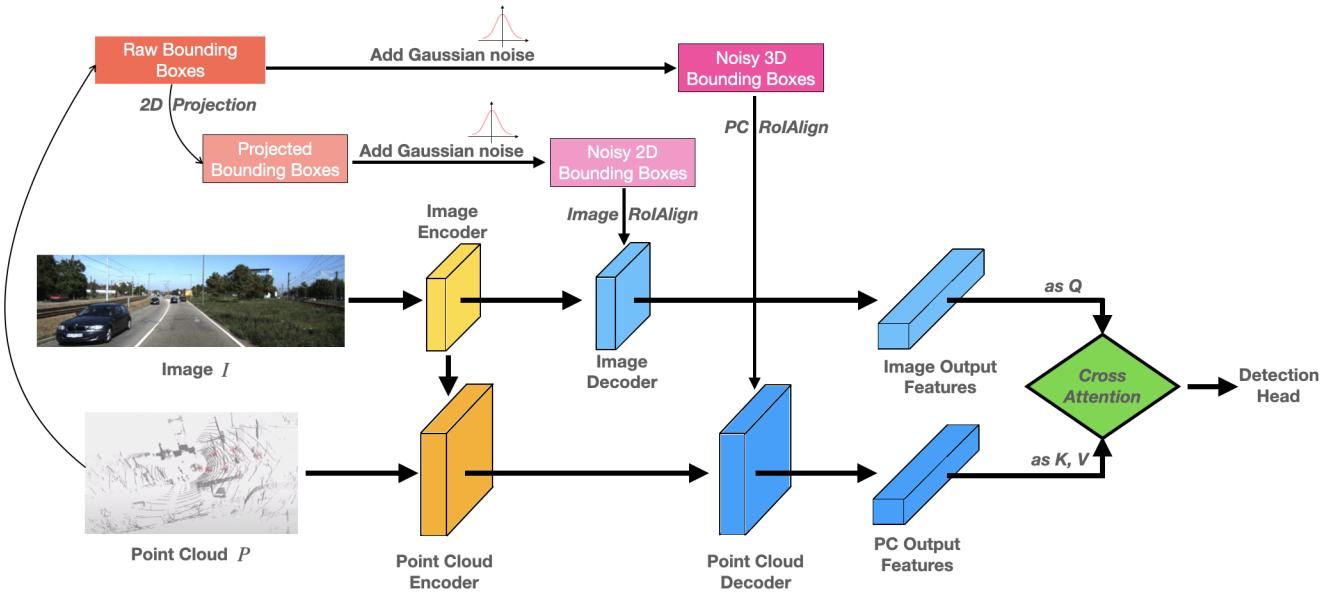

Figure 1. Overview of 3DifFusionDet.

the two modalities necessarily increases computing cost and inference time lag due to the complexity of LiDAR-based and camera-based detection systems. As a result, the problem of effectively fusing information from several modes still exists.

2.2. Diffusion Models

A diffusion model [31] is a generative model that gradually deconstructs observed data by introducing noise and restoring the original data by reversing this process. Diffusion models and denoising score matching are connected via denoising diffusion probabilistic models [30], which have recently sparked interest in applications of computer vision [7, 10, 26]. In several fields such as graph generation [32, 66], language understanding [4, 39], robust learning [46, 69] and temporal data modeling [35, 47] For instance, DDPMs have been used for super-resolution applications by SR3 [54].

2.3. Diffusion Models in Vision Tasks

Diffusion models have achieved great success in image generation and synthesis [16, 26, 30, 63]. Some pioneer works adopt the diffusion model for image segmentation tasks [1, 2, 7, 26]. Compared to these fields, their potential for object detection has yet to be fully explored. Previous approaches using a diffusion model for object detection are restricted to image-only input[10, 60]. Compared to 2D images, 3D LiDAR provides richer spatial information about objects, allowing for accurate depth perception and volumetric understanding, making it crucial in applications such as autonomous driving, where discerning the precise distance and orientation of surrounding vehicles is an essential aspect of safety. To the best of our knowledge, this is the first work that adopts a diffusion model to achieve 3D object detection.

3. Proposed Method

Notation and Terminology. In the remainder of this paper, we use upper or lower case letters (e.g., X or x) to represent scalars, lower case bold letters (e.g., $\mathbf{x}$) to denote column vectors, and bold-face upper case letters (e.g., $\mathbf{X}$) to represent matrices, and upper case calligraphic letters (e.g., $\mathcal{X}$) to indicate sets or higher-order tensors. We use $\mathbf{X}^\top$ and $\mathbf{x}^\top$ to represent the transpose of any matrix $\mathbf{X}$ and vector $\mathbf{x}$.

3.1. Preliminaries

Diffusion Models. Modern diffusion models often employ two Markov chains: a forward chain to corrupt the image using noise and a reverse chain to refine the noise back into an image [30]. Formally, the forward noise perturbing process at time step t is defined as $q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$ for a data distribution $\mathbf{x}_0 \sim q(\mathbf{x}_0)$. A variance schedule $\beta_1, \cdots, \beta_T$ is used to progressively add, customarily Gaussian, noise to the data:

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}\right). \quad (1)$$

Given $\mathbf{x}_0$, we can quickly generate a sample of $\mathbf{x}_t$ by sampling a Gaussian random vector $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and performing the following transformation:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + (1-\bar{\alpha}_t)\,\boldsymbol{\epsilon} \quad (2)$$

where $\bar{\alpha}_t = \prod_{s=0}^{t} (1 - \beta_s)$.

When being trained, a neural network is taught to forecast $\mathbf{x}_0$ from $\mathbf{x}_t$ for various $t \in \{1, \ldots, T\}$. When making inferences, we begin with random noise $\mathbf{x}_T$ and iteratively employ the reverse chain to produce $\mathbf{x}_0$.

DiffusionDet. It is the first diffusion model for the 2D object detection problem [10]. In their context, data samples are a collection of bounding boxes according to the formula $\mathbf{x}_0 = \mathbf{b}$, where $\mathbf{b} \in \mathcal{R}^{N \times 4}$ is a set of N boxes. DiffusionDet builds the diffusion process during training, then learns to reverse it, by training a neural network $f_\theta\left(\mathbf{x_t}, t\right)$ to predict $\boldsymbol{x}_0$ from $\boldsymbol{x}_t$ by minimizing the training objective with ℓ_2 loss [27]:

$$\mathcal{L}_{\text{train}} = \frac{1}{2} \left\| f_\theta\left(\mathbf{x}_t, t\right) - \mathbf{x}_0 \right\|^2. \tag{3}$$

The model can handle a fixed number of 2D instance boxes by padding additional boxes onto the initial ground truth boxes. DiffusionDet is optimized using set prediction loss [9] with optimum transport assignment [20] serving as the label assignment approach.

For the inference approach, DiffusionDet additionally makes use of DDIM [62] to improve the boxes for the subsequent iteration of the sampling process.

3.2. 3DifFusionDet Architecture

Framework Overview. Fig. 1 shows the overall architecture of the proposed 3DifFusionDet. It accepts multi-modal inputs, which include both RGB images and point clouds. The input images are defined as $\mathcal{I} \in \mathbb{R}^{H_I \times W_I \times 3}$, where H_I and W_I denote the image height and width dimensions, respectively. Meanwhile, the input point cloud is represented as a set of 3D points $\mathcal{P} \in \mathbb{R}^{H_P \times W_P \times D_P}$ in the $H_P \cdot W_P \cdot D_P$ 3D space, where each point is a vector of its (x, y, z)-coordinate.

We divide the entire model into feature extraction and feature decoding components for the same computationally intractable reason as DiffusionDet [10]: it would be (even more) difficult to directly apply f_θ on the raw 3D features at each iteration step. The feature extraction part runs only once to extract a deep feature representation from the raw input $\mathcal{X}$, while the feature decoding component takes this deep feature as a condition and trains to progressively draw the box predictions from noisy boxes $\mathbf{u_t}$.

To make full use of the complementary information provided by the two modalities, we separate encoders and decoders for each modality. In addition, we generate the noisy boxes $\mathbf{u_t^i}$ and $\mathbf{u_t^P}$ by distinct Gaussian distribution, using the diffusion model, training the image decoder and point cloud decoder individually to refine the 2D and 3D features. As for the connection of these two feature branches, simply concatenating them is going to incur an information cut which leads to reduced performance. To this end, a multi-head cross-attention mechanism [67] is introduced to deeply align these features. These aligned features are sent to the detection head to predict the final ground truth boxes without noise. We introduce those afore-mentioned functional components in detail to the reader:

Feature Extraction and Fusion Modules. Given raw images $\mathbf{I} \in \mathbb{R}^{H_I \times W_I \times 3}$ and 3D points $\mathcal{P} \in \mathbb{R}^{H_P \times W_P \times D_P}$, we use a separate feature extractor to encode them. For the image encoder, following [10], we try convolutional neural networks like ResNet [28] in the implementation of 3DifFusionDet. In light of [40], ResNet generated using a Feature Pyramid Network.

For the point encoder, we utilize voxel-based methods [14, 61] to extract, and adopt sparse-based methods [17, 18, 72] to process. Voxel-based methods convert LiDAR points to voxels. Compared to other families of point feature extraction methods, such as point-based methods, these methods discretize the point cloud into equally spaced 3D grids, reducing memory needs while keeping the original 3D shape information as much as possible. Sparsity-based processing methods further help networks to be computationally more efficient. These benefits balance out the diffusion model's comparably high computational requirements.

Compared to 2D features, 3D features contain an additional dimension, making learning more challenging. Considering this, besides feature extraction from raw modalities, we add a fusing path that adds the extracted image features as another input for the point encoder, promoting information exchange as well as exploiting learning from more diverse sources. Here we adopt the PointFusion strategy from [61], where points from the LiDAR sensor are projected onto the image plane. The concatenation of image features and the corresponding points are then jointly processed by the VoxelNet architecture.

Feature Decoders. The extracted image features $\mathbf{f^i}$ and the extracted point features $\mathbf{f^P}$ serve as input for the corresponding image and point decoders. Each decoder additionally incorporates input from the distinctively created noisy boxes, $\mathbf{u_t^i}$ or $\mathbf{u_t^P}$, to learn to refine the 2D and 3D features separately in addition to the corresponding extracted features. How these noisy boxes are created will be introduced in detail in Sec. 3.3.

The image decoder, taking inspiration from Sparse R-CNN [64], receives input from a collection of 2D proposal boxes to crop RoI-features [22] from feature maps created by the image encoder. The point decoder receives input from a collection of 3D proposal boxes to crop RoI features [12, 36, 59] from feature maps created by the image

encoder. For the point decoder, the input is a set of 3D proposal boxes to crop 3D RoI-feature from feature maps generated by the point encoder.

Cross-Attention Module. Following the decoding of the two feature branches, a method to combine them is needed. One straightforward way is to simply connect these two feature branches by concatenating them. This way appears too rough, which may lead the model to suffer from information cuts, resulting in reduced performance (shown in 4.3). Thus, a multi-head cross-attention mechanism (CA; [67]) is introduced to deeply align and refine these features, which is shown in Fig. 2.

Specifically, the output of the point decoder is treated as the source of $\mathbf{k}$ and $\mathbf{v}$, whereas the output of the image decoder is projected onto $\mathbf{q}$. The cross-attention process of the two-stream features is formulated as follows:

$$\mathrm{CA}(\mathbf{q}, \mathbf{k}, \mathbf{v}) = \mathrm{Attn}\left(\mathbf{q}\mathbf{w}_\mathbf{i}^\mathbf{q}, \mathbf{k}\mathbf{w}_\mathbf{i}^\mathbf{k}, \mathbf{v}\mathbf{w}_\mathbf{i}^\mathbf{v}\right)\mathbf{w}^{\text{out}} \tag{4}$$

where

$$\mathrm{Attn}(\mathbf{q}, \mathbf{k}, \mathbf{v}) = \mathrm{Softmax}\left(\frac{\mathbf{q}\mathbf{k}^T}{\sqrt{d_k}}\right)\mathbf{v} \tag{5}$$

such that $\mathbf{q}$, $\mathbf{k}$, and $\mathbf{v}$ are linearly projected to compute the attention matrix [67], and $\mathbf{q}_\mathbf{i}^\mathbf{q}$, $\mathbf{w}_\mathbf{i}^\mathbf{k}$ and $\mathbf{w}_\mathbf{i}^\mathbf{v}$ are the projection layers with shapes $\mathbb{R}^{d_{\text{model}} \times d_q}$, $\mathbb{R}^{d_{\text{model}} \times d_k}$ and $\mathbb{R}^{d_{\text{model}} \times d_v}$. Then, the refined BEV feature is obtained from the output layer $\mathbf{w}^{\text{Out}} \in \mathbb{R}^{d_v \times d_{\text{model}}}$. In addition, a residual path from the original point encoder is added to enhance feature propagation and strengthen the model's flexibility:

$$\mathbf{x} = \mathrm{CA}(\mathbf{q}, \mathbf{k}, \mathbf{v}) + \mathbf{x} \tag{6}$$

Seeking to learn more effectively, these aligned RoI features will be delivered to the detection head and produce results for bounding box regression and classification.

3.3. Training and Inference

Object Detection Head and Loss Function. Once getting the Transformer output, we add a bounding box head and a classification head to the output of the CA module. The regression output features can be represented as $\hat{\mathbf{u}}^i = (cx, cy, cz, l, w, h, \theta)$, while the classification head outputs $\hat{\mathbf{V}} \in \mathbb{R}^{N \times (C+1)}$ features, where C is the number of classes. One class is added for the "no object" class, indicating the absence of any recognized object. Each $\hat{\mathbf{v}}^i$ is the predicted probability of the object belonging to the positive class. We set the ground truth bounding box features to be $\mathbf{U}$ and ground truth class features to be $\mathbf{V}$, where each $\mathbf{v}^i$ is a one-hot encoder setting the component corresponding to the class label to be 1 while zeroing all other components.

The Hungarian set prediction loss is applied following [9, 10, 64]. In addition, by choosing the top k predictions with the lowest cost using OTA [20], an optimum transport assignment approach, we assign numerous predictions to each ground truth. For the cost computing on both of the Hungarian set prediction and OTA, we utilize Focal Loss [41] as classification cost, ℓ_1 Loss, Generalized IoU Loss [53] and Center Loss [79] as regression cost. Formally:

$$\mathcal{L}_{\text{Total}} = \lambda_1 \mathcal{L}_{\text{cls}} + \lambda_2 \mathcal{L}_{\text{reg}}. \tag{7}$$

The **classification loss** measures the error in predicting the object class label. Here, the Focal Loss function [41] is a modified version of the Cross-Entropy:

$$\mathcal{L}_{\text{cls}} = -\sum_i [\mathbf{v}^i (1-\hat{\mathbf{v}}^i)^\gamma \log(\hat{\mathbf{v}}^i) + (1-\mathbf{v}^i)\hat{\mathbf{v}}^{i\gamma} \log(1-\hat{\mathbf{v}}^i)], \tag{8}$$

where γ is a modulating factor that controls the weight given to each example. The **regression loss** measures the error in predicting the object's bounding box location (including center, size, and heading). Inspired by [79], we add the center loss $\mathcal{L}_{\text{center}}$ to jointly optimize for the best 3D bounding box estimation under the 3D IoU metric:

$$\mathcal{L}_{\text{reg}} = \lambda_3 \mathcal{L}_{\text{L1}} + \lambda_4 \mathcal{L}_{\text{Giou}} + \lambda_5 \mathcal{L}_{\text{Center}} \tag{9}$$

Training. During training, we build the diffusion process from ground truth to noise filters relying on the corresponding bounding boxes, teaching the model to reverse this procedure after the noise has been added. To perturb these ground truth boxes to noisy ones with Eq. (10), we randomly choose a time step t. Additionally, noisy box features are used to create noisy instance filters for training. The last step consists of padding the ground truth bounding boxes and their corruption as per [10]. Thus, we obtain the predicted 3D bounding boxes:

$$\begin{aligned} \mathbf{u}_t &= \sqrt{\bar{\alpha}_t}\mathbf{u}_0 + (1-\bar{\alpha}_t)\,\epsilon \\ \theta_0 &= \eta\left(f\left(\mathbf{u}_t, t\right)\right) \end{aligned} \tag{10}$$

where η denotes the 3D detection head layer and $f(\mathbf{u}, t))$ is the denoising process of the decoder.

Inference. Denoising sampling from noise to instance filters makes up the 3DifFusionDet inference pipeline. We first start with bounding boxes $\mathbf{u}_T$ sampled from a Gaussian distribution, where the model iteratively uses to refine its predictions as follows:

$$\begin{aligned} &\mathbf{u}_0 = f\left(\cdots\left(f\left(\mathbf{u}_{T-s}, T-s\right)\right)\right) \quad s = \{0, \cdots, T\}, \\ &\theta_0 = \eta\left(\mathbf{u}_0\right), \end{aligned} \tag{11}$$

Note that DDIM[62] is also used in our model, in line with DiffusionDet.

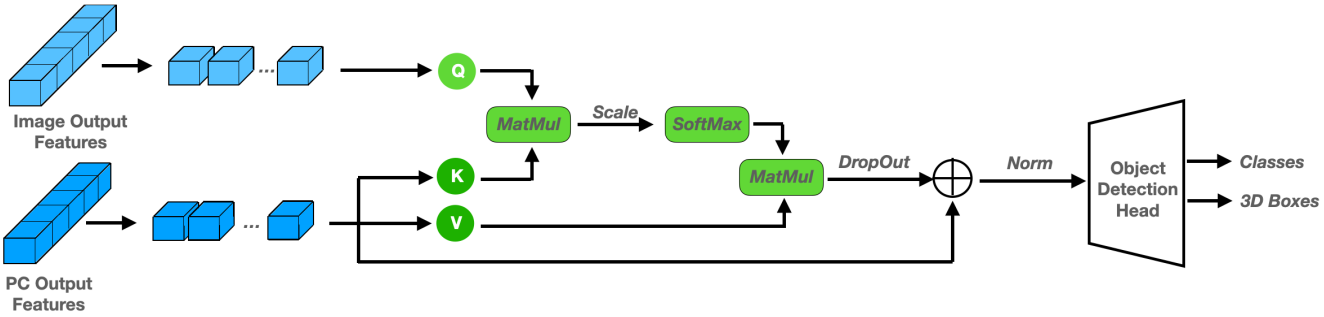

Figure 2. The Cross-Attention Module and Detection Head

4. Experiments

4.1. Experimental Setup

Datasets. To demonstrate the effectiveness of 3DifFusionDet, we present results on the KITTI 3D object detection benchmark [21]. It is divided into 7,481 training samples and 7,518 testing samples. The training samples are commonly divided into a training set (3,712 samples) and a validation set (3,769 samples) following [13], which we adopt. Before being fed into the models, the dataset is augmented by random 3D flips, random rotation, scale, and translation, and shuffled the point data. We compare 3DifFusionDet with existing methods on the test set by training our model on both the training and validation sets. We evaluate the validation set for ablation by training our model on only the train set.

Baselines. We compare our model to several baselines: PV-RCNN [58], MVX-Net [61], PointRCNN [57], Part-A^2-free [59], CT3D [56], and 3D-SSD [73]. All of them have been popular high-performance 3D object detection frameworks in recent years.

Implementation Details. We implement 3DifFusionDet using the MMDetection3D library [45]. The model is optimized using the Adam [34] optimizer with a learning rate of 0.0001, optimizer momentum $(\beta_1, \beta_2) = (0.9, 0.999)$, and a dropout rate of 0.3. We train the model on an NVIDIA RTX A6000 GPU for 60 epochs and validate after each epoch. The voxel grid is defined by a range and voxel size in 3D space. On KITTI, we use $[2, 46.8] \times [-30.08, 30.08] \times [-3, 1]$ for the range and $[0.16, 0.16, 0.16]$ for the voxel size for the x, y, and z axes, respectively. For the direct set prediction, we set the number of proposal boxes to 300. For the diffusion model, we use Gaussian diffusion with $D = 1000$. Following [9, 10, 79], we set the loss weights $\lambda_3 = 2.5$ while letting others be 1.

4.2. Results

We conduct experiments on the KITTI 3D object detection benchmark. Following the standard KITTI evaluation protocol (IoU $= 0.7$) for measuring the detection performance, Tab. 1 shows the mean average precision (mAP) scores for the 3DifFusionDet method compared to the state-of-the-art methods on the KITTI validation set using 3D and bird's eye view (BEV) evaluation. We report its performance for $D \in \{4, 8\}$, following[10, 26] and bold-face the two best-performing models for each task.

As Tab. 1, our approach shows significant performance improvements compared to the baselines. With $D = 4$, it is able to outperform most of the baselines with a relatively short inference time. By further increasing D such that $D = 8$, taking into account a higher inference time, we achieve the best performance among all models. This flexibility reveals a wide range of potential usages. The trade-off between accuracy and inference speed is discussed in Sec. 4.3.

4.3. Ablation Studies

Firstly, we show the efficiency of using **the fusion strategy**. We did two more experiments on KITTI that removed all feature passes from the Lidar modality and image modality, respectively, keeping only a single feature passageway. Showing the result in the first two lines of Tab. 2, we can see the performance could drop dramatically without the image feature or lidar feature fusing.

Secondly, we show the necessity of keeping the **Image RoI Align branch** and the **encoder feature fusion**. For designing a 3D Object detector from both Camera and Lidar using the diffusion model, the most straightforward approach should be directly applying the generated noisy 3D boxes as input to the fused 3D features. This way, however, could suffer from information cut, which leads to reduced performance, as shown in the third line of Tab. 2. Drawing upon it, besides putting the point cloud RoIAlign under the encoded 3D features, we also create the second branch that makes the image RoIAlign under the encoded 2D features.

Model	mAP_{BEV} (IoU $= 0.7$)			mAP_{3D} (IoU $= 0.7$)			Runtime
	Easy	Med	Hard	Easy	Med	Hard	(ms)
PV-RCNN[58]	86.2	84.8	78.7	N/A	N/A	N/A	59.2
MVX-Net (PF)[61]	89.5	84.9	79.0	85.5	73.3	67.4	125.6
PointRCNN[57]	87.9	90.2	85.5	88.6	88.9	77.4	100.1
CT3D[56]	88.5	86.1	79.0	90.5	87.1	79.0	N/A
3D-SSD[73]	89.7	89.5	78.7	89.4	87.5	78.4	38.9
Part-A^2-free[59]	88.0	90.2	85.9	89.0	88.5	78.4	80.8
OcTr[75]	89.5	82.4	77.3	87.3	75.5	75.4	63.6
FastPillars[77]	88.2	83.2	81.1	89.1	85.3	77.6	53.2
BEVFusion[42]	89.5	88.9	86.3	89.0	87.1	77.4	70.3
TED[70]	91.6	85.3	80.7	91.5	85.7	82.2	105.5
LoGoNet[38]	91.8	85.0	80.7	91.8	85.0	82.4	96.3
3DifFusionDet ($D = 4$)	89.9	91.3	85.3	90.5	88.2	79.7	43.2
3DifFusionDet ($D = 8$)	90.3	91.8	86.3	91.3	89.5	80.4	67.2

Table 1. Comparison of attained validation mAP (in %) on KITTI with IoU $= 0.7$.

	Easy	Med	Hard
W/o Lidar modality	48.4	41.3	38.6
W/o Image modality	67.3	56.7	52.4
W/o Image RoI Align	86.4	75.0	74.8
W/o encoder feature fusion	87.0	77.8	75.4
W/ Both	**90.5**	**88.2**	**79.7**

Table 2. Performance gap of removing certain parts on AP_{3D} (in %) on KITTI with IoU = 0.7

Fusion Align	Easy	Med	Hard
Sum	86.8	72.0	70.8
Concat	86.5	72.4	71.1
DP	86.2	80.5	75.6
MLP	87.0	83.9	76.5
CA	88.8	87.9	78.1
Res-CA	**90.5**	**88.2**	**79.7**

Table 3. Performance gap of different feature align strategies on AP_{3D} (in %) on KITTI with IoU = 0.7

#Boxes	D	Easy	Med	Hard	FPS
300	1	87.1	85.3	77.7	**19**
100	4	86.8	85.5	77.3	13
300	4	90.5	88.2	79.7	12
300	8	**91.3**	**89.5**	**80.4**	6

Table 4. Accuracy vs. speed. Using more proposal boxes incurs a higher performance gain at the cost of latency on AP_{3D} (IoU $=$ 0.7).

Much better utilization of the complementing information offered by the two modalities is shown in the fourth line of Tab. 2, by significantly higher performance.

Then we analyze the influence of using different **feature fusion strategies**: given the learned 2D and 3D represented features, how can we combine them more efficiently? Compared to 2D features, 3D features contain an additional dimension, making them more challenging to learn.

Firstly, we investigate the connection of the two feature branches following the decoding. Here we applied a multi-head cross-attention mechanism (CA and Res-CA) [67] to deeply align and refine these features. Besides this way, more straightforward ways like using the concatenation operation, the sum operation, the direct product operation (DP [8]), and using a multi-layer perceptron (MLP) are all investigated. The first four lines of Tab. 3 show their results. Among all, the cross-attention mechanism shows the best performance, with almost the same training and inference speed. This demonstrates its promising features.

Inspired by [61], we also add an information flow path from image feature to point feature by additionally projecting points from the LiDAR sensors, using the concatenation of image features and the corresponding points to be jointly processed by the VoxelNet architecture. The fifth and last line of Tab. 3 shows its benefits gain on detection accuracy.

Next, we investigate the **accuracy and inference speed trade-off**. By comparing the 3D detection accuracy and number of frames per second (FPS), we show the influence of choosing different proposal boxes as well as D. Shown in Tab. 4, the number of proposal boxes are chosen from $100, 300$, whereas D is chosen from $1, 4, 8$. The run time is evaluated on a single NVIDIA RTX A6000 GPU with a batch size of 1. We see that increasing the number of

proposal boxes from 100 to 300 significantly increases the accuracy gain with negligible latency cost (13 FPS vs. 12 FPS). On the other hand, better detection accuracy incurs a longer inference time. As we vary the D from 1 to 8, the 3D detection accuracy increases from steeply (Easy: 87.1 mAP to 90.5 mAP) to relatively slowly (Easy: 90.5 AP to 91.3 mAP), while the FPS decreases constantly. Note that we did the reference on only one RTX A6000 GPU. Referred to [26], who got nearby FPS values as ours by running on a single V100 GPU, arguing that their method could reach a similar real-time level as [10] in 2D object detections. Our model should also reach real-time inference speed when running on a high-performance GPU like A100, which [10] utilized.

4.4. Case Study and Future Work

Based on its unique properties, we discuss potential usages of our 3DifFusionDet. Generally, inferring accurately, robustly, and in real-time are three requirements in object detection tasks [43]. In the perception field of self-driving cars, considering the fact that a high-speed car needs to take extra time and distance to slow down or change direction due to inertia, the perception model is especially sensitive to the real-time requirement. More importantly, to guarantee a comfortable riding experience, the car should run as smoothly as possible with the minimum absolute value of acceleration, under the premise of safety. It is one primary advantage over other similar self-driving car products that withing smoother riding experiences. To this end, whether it is to speed up, slow down, or turn, self-driving cars should start to respond in a quick manner. The quicker a car starts to respond, the more robust space it earns for following operations and adjustments. This is even more important compared to getting the most precise detected object's classification or location first: as a car starts to respond, there is still time and distance to adjust its manners, which could be utilized to make further inferences in a more precise way, whose result subsequently finetunes the driving operation of the car.

Our 3DifFusionDet naturally matches the need. As Tab. 4 shows, when the inference step is small, the model could make inferences in a quick time with roughly high-accurate results. This initial perception is precise enough for a self-driving car to start its new response. As the inference step grows, higher accurate detected objects are generated, further finetuning its response. This *progressive detection manner* fits in nicely for this task. In addition, since our model could change the number of proposal boxes during reference, the prior information obtained from the small steps could in turn be used to optimize the real-time proposal box number. As Tab. 4 shows, the performance varies under different prior proposal boxes. Therefore, developing such a self-adaptive detector is one bright future work.

Besides self-driving cars, our model inherently matches any real-world scenario that requires short inferring time in continuous reaction space, especially in scenes where the detector is moving based on the detected result. Benefits from the nature of the diffusion model, 3DifFusionDet could find the almost-accurate real space regions of interest in a quick time, triggering the machine to start making new operations and self-optimization. The following higher-accurate perceptron further finetunes the machine's operations. To deploy our model into these moving detectors, one open issue is the strategy that combines the inferred information from between larger steps' early infers and smaller steps' latest infer, which is another open question.

5. Conclusion

This paper presents 3DifFusionDet, a novel 3D object detector with robust LiDAR and camera fusion. Formulating 3D object detection as a generative denoising process, it is the first work that applies a diffusion model to 3D object detection. This work investigates the most effective camera-LiDAR fusion alignment strategies within the context of the generative denoising process framework, and it proposes fusion align strategies to make full use of the complementing information offered by the two modalities. 3DifFusionDet achieves favorable performance compared to well-established detectors, demonstrating a promising future of diffusion models in object detection tasks. The robust learning results and flexible inference mode makes it promising in potential usages.

Acknowledgement

This work is partially supported by NSF through grants IIS-1763365 and IIS-2106972. We also thank the anonymous reviewers for their helpful feedback.

References

[1] Tomer Amit, Tal Shaharbany, Eliya Nachmani, and Lior Wolf. Segdiff: Image segmentation with diffusion probabilistic models, 2022. 3

[2] Emmanuel Brempong Asiedu, Simon Kornblith, Ting Chen, Niki Parmar, Matthias Minderer, and Mohammad Norouzi. Decoder denoising pretraining for semantic segmentation, 2022. 3

[3] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces, 2023. 2

[4] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces, 2023. 3

[5] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *2022*

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 18187–18197, 2022. 2

[6] Xuyang Bai, Zeyu Hu, Xinge Zhu, Qingqiu Huang, Yilun Chen, Hongbo Fu, and Chiew-Lan Tai. Transfusion: Robust lidar-camera fusion for 3d object detection with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1090–1099, 2022. 1, 2

[7] Dmitry Baranchuk, Ivan Rubachev, Andrey Voynov, Valentin Khrulkov, and Artem Babenko. Label-efficient semantic segmentation with diffusion models, 2022. 2, 3

[8] Bokai Cao, Hucheng Zhou, Guoqiang Li, and Philip S. Yu. Multi-view machines. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, page 427–436. Association for Computing Machinery, 2016. 7

[9] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers, 2020. 1, 4, 5, 6

[10] Shoufa Chen, Peize Sun, Yibing Song, and Ping Luo. DiffusionDet: Diffusion model for object detection. *arXiv preprint arXiv:2211.09788*, 2022. 1, 2, 3, 4, 5, 6, 8

[11] Ting Chen, Lala Li, Saurabh Saxena, Geoffrey Hinton, and David J. Fleet. A generalist framework for panoptic segmentation of images and videos, 2022. 2

[12] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving, 2016. 1, 2, 4

[13] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals using stereo imagery for accurate object class detection, 2017. 6

[14] Yukang Chen, Jianhui Liu, Xiangyu Zhang, Xiaojuan Qi, and Jiaya Jia. Voxelnext: Fully sparse voxelnet for 3d object detection and tracking, 2023. 1, 2, 4

[15] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving, 2022. 1

[16] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis, 2021. 2, 3

[17] Lue Fan, Ziqi Pang, Tianyuan Zhang, Yu-Xiong Wang, Hang Zhao, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Embracing single stride 3d object detector with sparse transformer, 2021. 4

[18] Lue Fan, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Fully sparse 3d object detection, 2022. 4

[19] Peng Gao, Minghang Zheng, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. Fast convergence of detr with spatially modulated co-attention, 2021. 1

[20] Zheng Ge, Songtao Liu, Zeming Li, Osamu Yoshie, and Jian Sun. Ota: Optimal transport assignment for object detection, 2021. 4, 5

[21] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012. 1, 2, 6

[22] Ross Girshick. Fast r-cnn, 2015. 4

[23] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, abs/1311.2524, 2013. 1

[24] Ruben Gomez-Ojeda, Jesus Briales, and Javier Gonzalez-Jimenez. Pl-svo: Semi-direct monocular visual odometry by combining points and line segments. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4211–4216, 2016. 1

[25] Alexandros Graikos, Nikolay Malkin, Nebojsa Jojic, and Dimitris Samaras. Diffusion models as plug-and-play priors, 2023. 1, 2

[26] Zhangxuan Gu, Haoxing Chen, Zhuoer Xu, Jun Lan, Changhua Meng, and Weiqiang Wang. Diffusioninst: Diffusion model for instance segmentation, 2022. 2, 3, 6, 8

[27] Amit Gupta and Siuwa M Lam. Weight decay backpropagation for noisy data. *Neural networks*, 11(6):1127–1138, 1998. 4

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 2, 4

[29] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn, 2017. 2

[30] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. 2, 3

[31] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 3

[32] Emiel Hoogeboom, Victor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d, 2022. 2, 3

[33] Peng Jin, Hao Li, Zesen Cheng, Kehan Li, Xiangyang Ji, Chang Liu, Li Yuan, and Jie Chen. Diffusionret: Generative text-video retrieval with diffusion model, 2023. 2

[34] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[35] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis, 2021. 3

[36] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven Waslander. Joint 3d proposal generation and object detection from view aggregation, 2017. 4

[37] Feng Li, Hao Zhang, Shilong Liu, Jian Guo, Lionel M. Ni, and Lei Zhang. Dn-detr: Accelerate detr training by introducing query denoising, 2022. 1

[38] Xin Li, Tao Ma, Yuenan Hou, Botian Shi, Yuchen Yang, Youquan Liu, Xingjiao Wu, Qin Chen, Yikang Li, Yu Qiao, and Liang He. Logonet: Towards accurate 3d object detection with local-to-global cross-modal fusion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 7

[39] Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B. Hashimoto. Diffusion-lm improves controllable text generation, 2022. 3

[40] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection, 2017. 4

[41] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2017. 5

[42] Zhijian Liu, Haotian Tang, Alexander Amini, Xinyu Yang, Huizi Mao, Daniela Rus, and Song Han. Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation, 2022. 2, 7

[43] Jiageng Mao, Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. 3d object detection for autonomous driving: A comprehensive survey, 2023. 1, 2, 8

[44] Ishan Misra, Rohit Girdhar, and Armand Joulin. An end-to-end transformer model for 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2906–2917, 2021. 2

[45] Contributors MMDetection3D. MMDetection3D: OpenMMLab next-generation platform for general 3D object detection. https://github.com/open-mmlab/mmdetection3d, 2020. 6

[46] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. Diffusion models for adversarial purification, 2022. 3

[47] Sung Woo Park, Kyungjae Lee, and Junseok Kwon. Neural markov controlled SDE: Stochastic optimization for continuous-time data. In *International Conference on Learning Representations*, 2022. 3

[48] Youngmin Park, Vincent Lepetit, and Woontack Woo. Multiple 3d object tracking for augmented reality. In *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, page 117–120, USA, 2008. IEEE Computer Society. 1

[49] Aditya Prakash, Kashyap Chitta, and Andreas Geiger. Multi-modal fusion transformer for end-to-end autonomous driving, 2021. 2

[50] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation, 2016. 1

[51] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum pointnets for 3d object detection from rgb-d data, 2017. 1

[52] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2015. 2

[53] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression, 2019. 5

[54] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J. Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement, 2021. 3

[55] Wenkang Shan, Zhenhua Liu, Xinfeng Zhang, Zhao Wang, Kai Han, Shanshe Wang, Siwei Ma, and Wen Gao. Diffusion-based 3d human pose estimation with multi-hypothesis aggregation, 2023. 2

[56] Hualian Sheng, Sijia Cai, Yuan Liu, Bing Deng, Jianqiang Huang, Xian-Sheng Hua, and Min-Jian Zhao. Improving 3d object detection with channel-wise transformer, 2021. 6, 7

[57] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud, 2018. 1, 6, 7

[58] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. PV-RCNN: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020. 6, 7

[59] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network, 2020. 1, 2, 4, 6, 7

[60] Yuguang Shi. Svdm: Single-view diffusion model for pseudo-stereo 3d object detection, 2023. 3

[61] Vishwanath A. Sindagi, Yin Zhou, and Oncel Tuzel. Mvx-net: Multimodal voxelnet for 3d object detection, 2019. 1, 2, 4, 6, 7

[62] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models, 2022. 4, 5

[63] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution, 2020. 3

[64] Peize Sun, Rufeng Zhang, Yi Jiang, Tao Kong, Chenfeng Xu, Wei Zhan, Masayoshi Tomizuka, Lei Li, Zehuan Yuan, Changhu Wang, and Ping Luo. Sparse R-CNN: end-to-end object detection with learnable proposals. *CoRR*, abs/2011.12450, 2020. 1, 4, 5

[65] Pei Sun, Mingxing Tan, Weiyue Wang, Chenxi Liu, Fei Xia, Zhaoqi Leng, and Dragomir Anguelov. Swformer: Sparse window transformer for 3d object detection in point clouds, 2022. 1

[66] Brian L. Trippe, Jason Yim, Doug Tischer, David Baker, Tamara Broderick, Regina Barzilay, and Tommi Jaakkola. Diffusion probabilistic modeling of protein backbones in 3d for the motif-scaffolding problem, 2023. 2, 3

[67] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017. 4, 5, 7

[68] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. 2

[69] Jinyi Wang, Zhaoyang Lyu, Dahua Lin, Bo Dai, and Hongfei Fu. Guided diffusion model for adversarial purification, 2022. 3

[70] Hai Wu, Chenglu Wen, Wei Li, Ruigang Yang, and Cheng Wang. Transformation-equivariant 3d object detection for autonomous driving. In *AAAI*, 2023. 7

[71] Yu Xiang, Wonhui Kim, Wei Chen, Jingwei Ji, Christopher Bongsoo Choy, Hao Su, Roozbeh Mottaghi, Leonidas J. Guibas, and Silvio Savarese. Objectnet3d: A large scale database for 3d object recognition. In *European Conference on Computer Vision*, 2016. 1

[72] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors (Basel, Switzerland)*, 18, 2018. 1, 4

[73] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3DSSD: Point-based 3d single stage object detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11040–11048, 2020. 1, 6, 7

[74] Yihan Zeng, Da Zhang, Chunwei Wang, Zhenwei Miao, Ting Liu, Xin Zhan, Dayang Hao, and Chao Ma. Lift: Learning 4d lidar image fusion transformer for 3d object detection. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17151–17160, 2022. 1, 2

[75] Chao Zhou, Yanan Zhang, Jiaxin Chen, and Di Huang. Octr: Octree-based transformer for 3d object detection, 2023. 7

[76] Shengchao Zhou, Weizhou Liu, Chen Hu, Shuchang Zhou, and Chao Ma. Unidistill: A universal cross-modality knowledge distillation framework for 3d object detection in bird's-eye view, 2023. 2

[77] Sifan Zhou, Zhi Tian, Xiangxiang Chu, Xinyu Zhang, Bo Zhang, Xiaobo Lu, Chengjian Feng, Zequn Jie, Patrick Yin Chiang, and Lin Ma. Fastpillars: A deployment-friendly pillar-based 3d detector, 2023. 7

[78] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection, 2017. 2

[79] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection, 2020. 1, 5, 6