A Simple Baseline for Multi-Camera 3D Object Detection

Yunpeng Zhang[1], **Wenzhao Zheng**[2], **Zheng Zhu**[1],
Guan Huang[1], **Jiwen Lu**[2], **Jie Zhou**[2]

[1]PhiGent Robotics
[2]Department of Automation, Tsinghua University
{yunpeng.zhang, guan.huang}@phigent.ai, zhengzhu@ieee.org,
zhengwz18@mails.tsinghua.edu.cn, {lujiwen, jzhou}@tsinghua.edu.cn

Abstract

3D object detection with surrounding cameras has been a promising direction for autonomous driving. In this paper, we present **SimMOD**, a **Sim**ple baseline for **M**ulti-camera **O**bject **D**etection, to solve the problem. To incorporate multi-view information as well as build upon previous efforts on monocular 3D object detection, the framework is built on sample-wise object proposals and designed to work in a two-stage manner. First, we extract multi-scale features and generate the perspective object proposals on each monocular image. Second, the multi-view proposals are aggregated and then iteratively refined with multi-view and multi-scale visual features in the DETR3D-style. The refined proposals are end-to-end decoded into the detection results. To further boost the performance, we incorporate the auxiliary branches alongside the proposal generation to enhance the feature learning. Also, we design the methods of target filtering and teacher forcing to promote the consistency of two-stage training. We conduct extensive experiments on the 3D object detection benchmark of nuScenes to demonstrate the effectiveness of SimMOD and achieve competitive performance. Code will be available at https://github.com/zhangyp15/SimMOD.

Introduction

The safety of self-driving vehicles strongly depends on an accurate and comprehensive understanding of the surroundings. As the core task of 3D scene understanding, 3D object detection aims to produce a 3D bounding box for each concerned object around the ego vehicle. Existing methods for 3D object detection usually rely on LiDAR sensors (Lang et al. 2019; Shi, Wang, and Li 2019) or stereo cameras (Qin, Wang, and Lu 2019; Sun et al. 2020a) to provide explicit distance information, but the high cost of LiDAR and the instability of stereo systems have hindered the practical application of autonomous driving. Therefore, monocular 3D object detection has been considered a promising direction and received extensive attention in recent years.

While most existing methods (Chen et al. 2016; Wang et al. 2021a; Ma et al. 2021; Zhang, Lu, and Zhou 2021) focus on generating 3D bounding boxes based on one single monocular image, it is undoubtedly beneficial to incorporate the information from multiple surrounding cameras (Wang et al. 2021b; Huang et al. 2021), which are commonly available on modern self-driving vehicles as well as provided in recently released driving datasets (Caesar et al. 2020; Sun et al. 2020b). To fully exploit the multi-view images, DETR3D (Wang et al. 2021b) employs a set of learned queries to adaptively integrate multi-scale features by geometric projection. After the iterative processing, these queries are decoded into a set of 3D bounding boxes. However, DETR3D suffers from slow convergence and requires specialized pretraining to achieve promising performance.

In this paper, we present SimMOD, a simple baseline for multi-camera 3D object detection, to serve as an effective and highly-extensible method to facilitate the development of multi-camera detection. SimMOD is formulated as a two-stage propose-and-fuse framework, which is compatible with existing methods for monocular 3D object detection and can also integrate information from multiple cameras. Specifically, we first utilize the image encoder to construct multi-scale feature maps for the multi-view images. The proposal head is then applied to distinguish the foreground objects and generate the sample-wise proposals with visual features, initial 3D positions, and view-level-aware encodings. In addition to the essential branches like the foreground classification and position estimation, we further incorporate a series of auxiliary branches to predict various visual attributes during the training stage. These auxiliary branches can significantly improve the learning of discriminative features. After the proposals are generated in every perspective view, we transform all proposals into the uniform ego-car coordinates and employ an iterative detection head to refine these proposals. This process includes the inter-proposal attention mechanism for interactions and the geometry-based feature sampling for aggregating multi-view information. Finally, the refined proposals are used to generate the detection results, which are supervised by the set-based losses to enable end-to-end predictions. Since the incomplete recall of generated proposals can hurt the bipartite matching, we also propose two techniques to provide consistent supervision. As shown in Fig. 1, the sample-wise object proposals can focus on the interested objects, while most queries from DETR3D are scattered around the background regions.

With extensive experiments, we verify the effectiveness of our proposed method on the nuScenes (Caesar et al. 2020) dataset. Compared with the baseline DETR3D, SimMOD

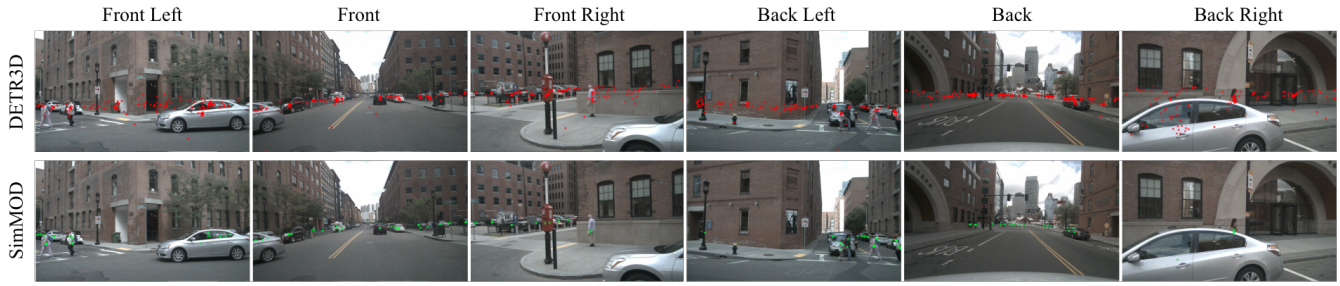

Figure 1: Qualitative comparison between fixed queries and sample-wise proposals. As shown in the first row, the learned queries from DETR3D are scattered around the image and most lie in the background regions. In contrast, our sample-wise proposals (the second row) can effectively attend to the interested objects. (Best viewed in color.)

boosts the performance by 5.9% NDS with ResNet-50 and 3.0% NDS with ResNet-101.

Related Work

Image-based 2D Object Detection

Modern image-based methods for 2D object detection can include two-stage and one-stage detectors. As the representation of two-stage methods, Faster-RCNN (Ren et al. 2015) uses the region proposal network to generate proposals and refine them with the RCNN (Girshick et al. 2014) head. By contrast, one-stage methods, like YOLO (Redmon et al. 2016) and SSD (Liu et al. 2016), generate dense predictions in one shot. To eliminate the requirement of heuristic post-processing, the recent DETR (Carion et al. 2020) provides an end-to-end detection framework based on transformer (Vaswani et al. 2017) and set-based losses. Despite its graceful design, DETR suffers from slow convergence and limited performance. To this end, Deformable DETR (Zhu et al. 2021) proposes the deformable attention to efficiently utilize multi-scale features, while TSP (Sun et al. 2021) combines FCOS and the transformer-based set prediction for two-stage detection. Motivated by the two-stage DETR variants for 2D object detection, we present our two-stage framework for multi-camera 3D object detection. In the first stage, we recognize foreground objects in each perspective view and generate object proposals. For the second-stage refinement, we lift these proposals to the uniform 3D space and incorporate multi-view information for refinement and prediction.

Monocular 3D Object Detection

With one single image as input, monocular 3D object detection aims to predict the 3D bounding boxes and categories of the interested objects. A series of methods rely on the additional depth information to solve the problem, including pseudo-LiDAR (Wang et al. 2019), D4LCN (Ding et al. 2020), and CaDDN (Reading et al. 2021). Other methods directly predict the 3D locations (Ma et al. 2021; Zhang, Lu, and Zhou 2021) or incorporate the geometric clues (Mousavian et al. 2017; Wang et al. 2022). Recently, FCOS3D (Wang et al. 2021a) extends the 2D detector FCOS (Tian et al. 2019) for monocular 3D object detection and reports the performance on the multi-camera nuScenes dataset (Caesar et al. 2020). When dealing with multi-camera systems, monocular methods separately process each image and merge the results of detection through heuristic post-processing. The formulation is inherently sub-optimal because it fails in utilizing multi-view information and cannot benefit from end-to-end training.

Multi-camera 3D Object Detection

Recent methods for multi-camera 3D object detection decompose into two streams. A line of methods (Huang et al. 2021; Li et al. 2022; Zhang et al. 2022) focuses on learning the Bird-Eye-View representations from images and then detects objects in BEV. Though these methods can construct the dense BEV representations for wide extensions, they can suffer from the accumulated errors in the view transformation. The other stream of methods still works on the level of objects. DETR3D (Wang et al. 2021b) learns a set of object queries, which can extract multi-camera features through the geometric projections of their reference points. However, the object queries of DETR3D are simply parameterized embeddings and cannot flexibly adapt to the different input samples. In this paper, we take DETR3D as our baseline and further propose a two-stage framework where high-quality and sample-dependent proposals are first generated and then refined with multi-view features in the second stage. We demonstrate the proposed method greatly outperforms the baseline DETR3D at the convergence rate and detection performance.

Approach

Overview

Formally, the input of the overall framework includes a set of M surrounding images $\mathcal{I} = \{\mathbf{I}_{1:M}\} \subset \mathbb{R}^{\mathrm{H}\times\mathrm{W}\times 3}$ and their corresponding camera calibration matrices $\mathcal{K} = \{\mathbf{K}_{1:M}\} \subset \mathbb{R}^{3\times 3}$ and $\mathcal{T} = \{\mathbf{T}_{1:M}\} \subset \mathbb{R}^{4\times 4}$. The matrix $\mathbf{K}_j$ refers to the intrinsic matrix of camera j and $\mathbf{T}_j$ refers to the transformation matrix from the coordinate in camera j to the top-view LiDAR coordinate. As the learning targets, the ground-truth 3D bounding boxes are defined as $\mathcal{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_N\} \subset \mathbb{R}^9$ and their categorical labels are denoted as $\mathcal{C} = \{c_1, \ldots, c_N\} \subset \mathcal{Z}$. Each box $\mathbf{b} = (x, y, z, w, l, h, \theta, v_x, v_y)$ contains the position, size, yaw, and velocity. The number of categories is denoted as N_c

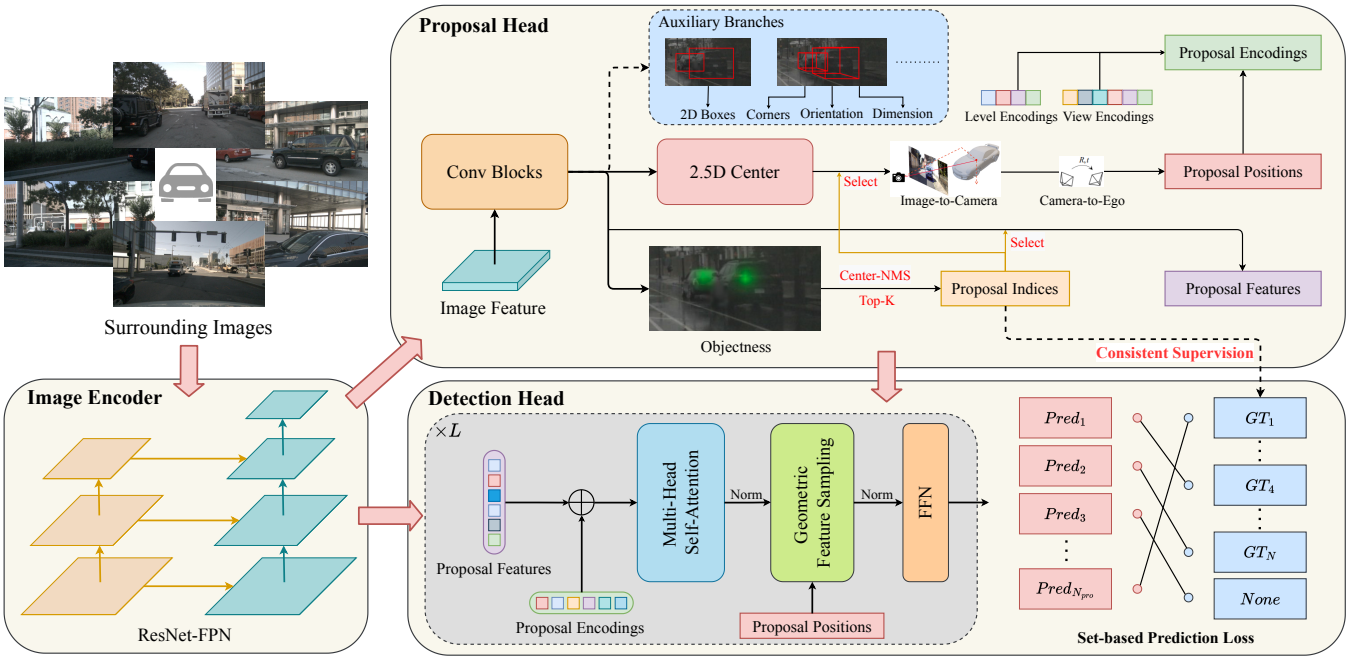

Figure 2: The overall framework of SimMOD. With the surrounding images as input, SimMOD first extracts multi-scale feature maps with the image encoder, which consists of the backbone and the feature pyramid network. Next, the proposal head processes each feature map to generate the object proposals, including the features, positions, and encodings. Finally, the multi-view and multi-scale proposals are collected in the ego-car coordinates and iteratively refined. The set-based detection loss is applied for end-to-end predictions. (Best viewed in color.)

and $N_c = 10$ for the nuScenes detection benchmark (Caesar et al. 2020). For convenience, we use $\mathcal{H}$ to transform the normal coordinate into the homogeneous coordinate and $\mathcal{H}^{-1}$ for the inverse. As illustrated in Fig. 2, the proposed framework first extracts multi-scale features from the input multi-view images. Then the proposal head generates multi-view proposals and lifts them into the uniform 3D space. Finally, the detection head iteratively aggregates multi-view information and refines the proposals for the final prediction. The model is end-to-end optimized with two-stage and consistent supervision.

Image Encoder

For a fair comparison, we follow DETR3D (Wang et al. 2021b) to build the image-view encoder with the classical ResNet (He et al. 2016) and FPN (Lin et al. 2017a). The deformable convolutions (Dai et al. 2017) are also applied in the last two stages of the backbone for a better trade-off between accuracy and efficiency. Considering the variable scales of interested objects, the encoder can generate multi-scale feature maps for the generation and refinement of object proposals. Specifically, we use four levels with strides (8, 16, 32, 64) for the experiments.

Proposal Head

The proposal head consists of four shared convolutional blocks and multiple small heads for different targets. Note that the proposal head is shared across different levels of feature maps. As for the distribution of objects to different feature levels and points, we follow the same settings as FCOS3D (Wang et al. 2021a). The proposal head is intended to distinguish foreground objects in the image domain, produce initial guesses about the 3D positions of these objects, and provide effective features for further refinement. We first introduce the generation of proposals and then elaborate on the design of proposal encodings. Finally, the auxiliary branches are incorporated to improve feature learning.

Proposal generation. The branches for generating proposals include classification, centerness, offset, and depth. The first two branches are combined to determine whether a point on the feature map corresponds to certain object centers, while the last two branches predict the 2.5D center, which can be lifted into the 3D position with camera matrices. Formally, we denote the predicted classification distribution as $\mathbf{P}_{cls} \in \mathbb{R}^{N_c \times H' \times W'}$ and centerness as $\mathbf{P}_{ctr} \in \mathbb{R}^{H' \times W'}$, where (H', W') refers to sizes of the specific feature map. The objectness map $\mathbf{P}_{obj} \in \mathbb{R}^{H' \times W'}$ can be computed as

$$\mathbf{P}_{obj} = \max\left(\mathbf{P}_{cls}, dim = 0\right) \odot \mathbf{P}_{ctr} \tag{1}$$

where $\odot$ refers to the Hadamard product. Then we apply a 3×3 max-pooling on the objectness map to remove duplicates. Finally, the N_{pro} top-scored positions are selected as object proposals. For each proposal, we assume it lies at camera j and denote its corresponding pixel on the input image as (u, v), the predicted offset as $(\Delta u, \Delta v)$, and the depth

as d. The proposal position (x, y, z) is computed as

$$(x, y, z, 1)^T = \mathbf{T}_j \mathcal{H} \left({\mathbf{K}_j}^{-1} \left[u + \Delta u, v + \Delta v, 1 \right]^T * d \right) \tag{2}$$

where the image-view proposal is first lifted into the camera coordinate and then transformed to the top-view ego coordinate. The function $\mathcal{H}$ is used to get the homogeneous coordinate. In this way, the perspective proposals produced from each surrounding image are lifted into the same 3D coordinate system. We use $\mathbf{X}_{pro} \in \mathbb{R}^{N_{pro} \times 3}$ to denote the proposal positions. To further extract the proposal features for later refinement, we concatenate the corresponding features at the classification and regression branches and reduce the channels to get the proposal features $\mathbf{F}_{pro} \in \mathbb{R}^{N_{pro} \times C}$, where C is the number of feature channels.

Proposal encodings. To further embed the positional information, we incorporate the corresponding camera-view and feature-level with the proposal positions to produce the proposal encodings for the subsequent refinement. Specifically, we define the learnable level encodings $\mathbf{E}_{level} \in \mathbb{R}^{N_{level} \times C}$ and view encodings $\mathbf{E}_{view} \in \mathbb{R}^{M \times C}$, where N_{level} and M refer to the numbers of feature levels and surrounding cameras. With the generated proposals, we select their corresponding level encodings and view encodings, concatenate them with the proposal positions, and linearly project to the proposal encodings $\mathbf{E}_{pro} \in \mathbb{R}^{N_{pro} \times 3}$.

Auxiliary branches. The auxiliary branches aim to improve the object-awareness of preceding features by regressing other visual attributes of the objects. Since these branches only exist during the training stage, the computational burden for inference is not increased. We include the regression of the 2D bounding box, eight projected corners of the 3D bounding box, rotation, 3D size, and velocity as the auxiliary tasks. The supervision from these extra tasks not only improves the perception of fine-grained details like object boundaries and key-points, but also forces the network to learn high-level 3D information.

Detection Head

Given the multi-scale features and multi-view object proposals, the detection head motivated by DETR3D (Wang et al. 2021b) is designed to iteratively refine the proposals and generate predictions. Analogously to (Carion et al. 2020; Wang et al. 2021b), the detection head includes L iterative layers. Each layer contains the interaction among proposals and the interaction between proposals and image features. Formally, we use $\mathbf{F}_I^{jk}$ to represent the k_{th} level feature map from the image of camera j. The proposal features and positions are denoted as $\mathbf{F}_{pro}$ and $\mathbf{X}_{pro}$. Three steps during each iteration are introduced in the following paragraphs.

Interaction between proposals and images. The interaction between proposals and images is important in two aspects. On the one hand, proposals generated from single-view images can further access the visual features from other views. On the other hand, the proposal features should be updated with the refinement of proposal positions during the iterations. To sample relevant features from multi-view and multi-scale feature maps, we project the proposal positions onto all these feature maps and use bi-linear interpolation to derive corresponding features. We use $\mathbf{f}_I^{jk}$ to represent the sampled features at $\mathbf{F}_I^{jk}$, and σ^{jk} for the binary indicator to reflect whether the projected pixel is valid. Then the aggregated feature is given by:

$$\mathbf{f}'_{pro} = \mathbf{f}_{pro} + \frac{1}{\sum_j \sum_k \sigma^{jk}} \sum_j \sum_k \sigma^{jk} \mathbf{f}_I^{jk} \tag{3}$$

Applying the aggregation in Eq. (3) for every proposal can get the updated proposal features $\mathbf{F}'_{pro}$.

Interaction among proposals. The interaction among proposals is essential for removing duplicates and enabling end-to-end object detection. The multi-head attention mechanism (Vaswani et al. 2017) is utilized to realize the interaction. Specifically, we first compute the proposal encodings $\mathbf{E}_{pro} \in \mathbb{R}^{N_{pro} \times C}$ to incorporate the information of positions, feature levels, and camera views. Then we add these encodings to the proposal features and linearly project them to the queries $\mathbf{Q}_{pro}$, keys $\mathbf{K}_{pro}$, and values $\mathbf{V}_{pro}$. Finally, the attentive interaction is formulated as in Eq. (4):

$$\mathbf{F}''_{pro} = \mathbf{F}'_{pro} + \text{softmax} \left(\frac{\mathbf{Q}_{pro} \mathbf{K}_{pro}^T}{\sqrt{C}} \right) \mathbf{V}_{pro} \tag{4}$$

Decoding and updating proposals. At the end of each iteration, we predict the 3D bounding box $\hat{\mathbf{b}}_i$ and its categorical distribution $\hat{\mathbf{c}}_i$ with two linear layers. The regression target of $\hat{\mathbf{b}}_i$ is defined as $(\Delta x, \Delta y, \Delta z, w, l, h, \sin\theta, \cos\theta, v_x, v_y)$, where $(\Delta x, \Delta y, \Delta z)$ is the residual vector between current proposal positions and their matched ground-truth boxes. With proposal features updated with the above interactions, we also refine the proposal positions by adding the predicted residual vectors.

Consistent Supervision

During the training stage, it is understandable that the generated proposals cannot achieve a complete recall of all ground-truth objects. Since the DETR-style set prediction loss is based on the bipartite matching between the refined proposals and the target objects, those objects missed by the proposals can hurt the matching and bring harmful gradients. To ensure the consistency between proposals and targets, we propose the following two techniques, including target filtering and teacher forcing.

Target filtering. A natural way to maintain consistency is to remove the missed target objects for the bipartite matching. For each proposal, it can be matched to a target object if the proposal lies within the 2D bounding box. Then we compute the union of the matched objects from all proposals, which serves as the supervision for the second stage.

Teacher forcing. When training the recurrent neural networks, the method of teacher forcing (Bengio et al. 2015) randomly uses the ground-truth state, rather than the predicted state, as the next input, which can avoid the propagation of errors and stabilize the early training. To this end, we

Method	Backbone	NDS ↑	mAP ↑	mATE ↓	mASE ↓	mAOE ↓	mAVE ↓	mAAE ↓
FCOS3D (Wang et al. 2021a)	R101	0.373	0.299	0.785	0.268	0.557	1.396	**0.154**
DETR3D (Wang et al. 2021b)		0.374	0.303	0.860	0.278	0.437	0.967	0.235
PGD (Wang et al. 2022)		0.409	0.336	0.732	**0.263**	0.423	1.285	0.172
BEVDet (Huang et al. 2021)		0.396	0.330	**0.702**	0.272	0.534	0.932	0.251
Ego3RT (Lu et al. 2022)		0.409	0.355	0.714	0.275	0.421	0.988	0.292
PETR (Liu et al. 2022)		0.421	**0.357**	0.710	0.270	0.490	0.885	0.224
SimMOD		**0.435**	0.351	0.717	0.267	**0.388**	**0.849**	0.187
FCOS3D† (Wang et al. 2021a)	R101	0.393	0.321	0.746	0.265	0.503	1.351	**0.160**
PGD† (Wang et al. 2022)		0.425	0.358	0.667	**0.264**	0.435	1.278	0.177
DETR3D† (Wang et al. 2021b)		0.425	0.346	0.773	0.268	0.383	0.842	0.216
Ego3RT† (Lu et al. 2022)		0.450	**0.375**	**0.657**	0.268	0.391	0.850	0.206
PolarDETR† (Chen et al. 2022a)		0.444	0.365	0.742	0.269	0.350	0.829	0.197
BEVFormer-S† (Li et al. 2022)		0.448	**0.375**	0.725	0.272	0.391	0.802	0.200
PETR† (Liu et al. 2022)		0.442	0.370	0.711	0.267	0.383	0.865	0.190
SimMOD†		**0.455**	0.366	0.698	**0.264**	**0.340**	**0.784**	0.197

Table 1: Comparison to state-of-the-arts on the nuScenes validation set. †: initialized from a FCOS3D checkpoint.

Method	NDS ↑	mAP ↑	mATE ↓	mAOE ↓
FCOS3D	0.317	0.213	0.841	0.604
DETR3D	0.356	0.231	0.825	**0.400**
SimMOD	**0.394**	**0.274**	**0.789**	0.437
FCOS3D†	0.329	0.229	0.816	0.571
DETR3D†	0.384	0.268	0.807	0.453
SimMOD†	**0.424**	**0.297**	**0.725**	**0.325**

Table 2: Comparisons in the overlap regions. All methods use ResNet-101 as the backbone. †: initialized from a FCOS3D checkpoint.

Method	Backbone	NDS	mAP	mATE	mAOE
CenterNet	DLA34	0.328	**0.306**	**0.716**	0.609
SimMOD	DLA34	**0.386**	0.294	0.780	**0.468**
DETR3D	R50	0.373	0.302	0.811	0.493
SimMOD	R50	**0.432**	**0.339**	**0.727**	**0.356**
FCOS3D	R101	0.373	0.299	0.785	0.557
DETR3D	R101	0.374	0.303	0.860	0.437
SimMOD	R101	**0.435**	**0.351**	**0.717**	**0.388**
DETR3D†	R101	0.425	0.346	0.773	0.383
SimMOD†	R101	**0.455**	**0.366**	**0.698**	**0.340**

Table 3: The scalability of SimMOD with different image backbones. †: initialized from a FCOS3D checkpoint.

are motivated to randomly replace the predicted objectness map with the ground-truth so that all annotated objects can have their corresponding proposals.

Loss Functions

Loss for proposal head. We follow the loss functions in FCOS3D (Wang et al. 2021a) for training the proposal head. The classification branch is optimized with focal loss (Lin et al. 2017b), while the centerness branch is trained with binary cross-entropy loss. All other regression branches, including the offset, depth, 2D bounding box, size, and orientation, are supervised with smooth L_1 loss. The integral loss for proposal generation is denoted as $\mathcal{L}_{pro}$.

Loss for detection head. Following DETR3D, the discrepancy between the predicted and ground-truth objects is quantified as the set prediction loss for supervision. Without loss of generality, the ground-truth objects are denoted as $\mathcal{B} = \{\mathbf{b}_i\}_{i=1}^{N}$ and their categorical labels as $\mathcal{C} = \{c_i\}_{i=1}^{N}$. Also, the predicted bounding boxes and categorical distributions are represented with $\hat{\mathcal{B}} = \{\hat{\mathbf{b}}_i\}_{i=1}^{N_{pro}}$ and $\hat{\mathcal{C}} = \{\hat{\mathbf{c}}_i\}_{i=1}^{N_{pro}}$. We pad the ground-truth boxes with $\varnothing$ up to N_{pro} objects and use the Hungarian algorithm to compute the optimal assignment. Formally, finding the optimal bipartite matching requires searching for the optimal permutation $\hat{\sigma}$ of N_{pro} elements with the lowest matching cost:

$$\hat{\sigma} = \arg\min_{\sigma} \sum_{i=1}^{N_{pro}} \mathcal{L}_{\text{match}} \left(\mathbf{b}_i, c_i, \hat{\mathbf{b}}_{\sigma(i)}, \hat{\mathbf{c}}_{\sigma(i)} \right) \quad (5)$$

We define the pair-wise matching cost $\mathcal{L}_{\text{match}}$ to consider both the classification and regression, as detailed in Eq. (6):

$$\mathcal{L}_{\text{match}} = -1_{\{c_i \neq \varnothing\}} \hat{\mathbf{c}}_{\sigma(i)}(c_i) + 1_{\{c_i \neq \varnothing\}} |\mathbf{b}_i - \hat{\mathbf{b}}_{\sigma(i)}| \quad (6)$$

When the optimal assignment $\hat{\sigma}$ is computed, the set prediction loss is formulated as

$$\mathcal{L}_{\text{det}} = \sum_{i=1}^{N_{pro}} -\log \hat{\mathbf{c}}_{\hat{\sigma}(i)}(c_i) + 1_{\{c_i \neq \varnothing\}} |\mathbf{b}_i - \hat{\mathbf{b}}_{\hat{\sigma}(i)}| \quad (7)$$

Total loss. The whole framework is end-to-end optimized with the two-stage loss $\mathcal{L} = \lambda \mathcal{L}_{pro} + \mathcal{L}_{\text{det}}$, where $\lambda = 1.0$ for our experiments. During the training stage, we follow existing methods (Carion et al. 2020; Wang et al. 2021b) to deeply supervise the intermediate predictions from every layer of iteration. For inference, we only use the predictions from the last layer.

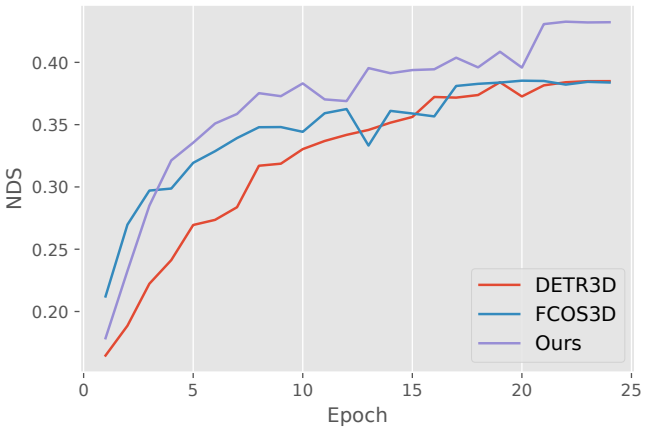

Figure 3: The intermediate performance of existing methods and our proposed SimMOD on the nuScenes validation set.

Method	NDS ↑	mAP ↑	mATE ↓	mAOE ↓
Fixed	0.362	0.293	0.855	0.502
Proposal	0.401	0.347	0.742	0.479
w. center-NMS	0.409	0.349	0.726	0.487
w. auxi (SimMOD)	**0.437**	**0.349**	**0.725**	**0.392**

Table 4: Ablation studies on sample-wise proposals.

Filter	Teacher	NDS ↑	mAP ↑	mATE ↓	mAOE ↓
		0.427	0.343	0.728	0.425
✓		0.436	0.347	**0.724**	**0.383**
	✓	0.433	0.346	0.737	0.400
✓	✓	**0.437**	**0.349**	0.725	0.392

Table 5: Ablation studies on the consistent supervision.

Experiments

Implementation Details

Dataset. We conduct extensive experiments on the large-scale autonomous driving dataset nuScenes (Caesar et al. 2020) to evaluate the proposed method. The nuScenes dataset provides 1000 sequences of different scenes collected in Boston and Singapore. Each sequence is about 20 seconds long and annotated with accurate 3D bounding boxes at 2Hz, contributing to a total of 1.4M object bounding boxes. These sequences are officially split into 700/150/150 ones for training, validation, and testing. For each sample, the images from six surrounding cameras and their camera calibrations are used as the input.

Evaluation metrics. Following the official evaluation protocol, the metrics include mean Average Precision (mAP) and a set of True Positive (TP) metrics, which contains the average translation error (ATE), average scale error (ASE), average orientation error (AOE), average velocity error (AVE), and average attribute error (AAE). Finally, the nuScenes detection score (NDS) is defined to consolidate the above metrics by computing a weighted sum as in Eq. (8):

$$\text{NDS} = \frac{1}{10}\left[5\text{mAP} + \sum_{\text{mTP}\in\mathcal{TP}} (1 - \min(1, \text{mTP}))\right] \quad (8)$$

Network architectures. Unless specified, we use the ImageNet-pretrained ResNet-101 (He et al. 2016) as the backbone network. We use the feature channel of 256 for multi-scale feature maps and proposal features. Within the proposal head, we define learnable and level-wise scale factors for regressing the offsets, depths, 2D boxes, and corners. We use the top-scored 600 proposals and filter low-scored proposals. The detection head contains six layers for iterative refinement.

Training and inference. SimMOD is implemented based on mmdetection3d. The model is end-to-end trained with AdamW (Loshchilov and Hutter 2019) optimizer for 24 epochs, with the initial learning rate as 2e-4 and weight decay as 0.01. Multi-step learning rate decay is applied. The input resolution is 1600 × 900. We train the model on 8 NVIDIA GeForce RTX 3090 GPUs with per-GPU batch size as 1. Following DETR3D (Wang et al. 2021b), we use color distortion and GridMask (Chen et al. 2020) for data augmentation.

Benchmark Results

As shown in Tab. 1, we conduct a comprehensive comparison between the proposed SimMOD with existing state-of-the-art methods on the nuScenes validation set. When using the ImageNet-pretrained R101 as the backbone network, SimMOD with dynamic proposals greatly improves the baseline DETR3D by 6.1% NDS and outperforms all existing methods. When initialized from the pretrained FCOS3D checkpoint, SimMOD achieves 45.5% NDS and is still 3.0% NDS higher than the baseline DETR3D. Despite the simplicity, the two-stage propose-and-fuse framework achieves the new state-of-the-art performance on the nuScenes validation set. Considering the inference speed, SimMOD can perform 2.3 FPS and is slightly faster than the baseline DETR3D with 2.0 FPS.

Performance in Overlap Regions

With only limited information, it is difficult to localize the truncated objects at the border regions of images. As shown in Tab. 2, the early-fusion methods, including DETR3D and our SimMOD, can effectively utilize the multi-view information and outperform the monocular method FCOS3D. When initialized from the FCOS3D checkpoint, SimMOD with sample-wise proposals further improves the baseline DETR3D by 4.0% NDS and 2.9% mAP.

Scalability with Different Backbones

In Tab. 3, we investigate the scalability of SimMOD with different backbone networks, including DLA (Yu et al. 2018) and various ResNets (He et al. 2016). We can observe that SimMOD notably outperforms the baseline model under every backbone setting. Also, SimMOD with R50 achieves 43.2% NDS and can outperform DETR3D with R101 (42.5% NDS).

View	Level	NDS ↑	mAP ↑	mATE ↓	mAOE ↓
		0.424	0.347	0.740	**0.388**
✓		0.434	0.347	**0.718**	0.401
	✓	0.432	**0.351**	0.728	0.396
✓	✓	**0.437**	0.349	0.725	0.392

Table 6: Ablation studies on the proposal encoding.

Proposal Weight λ	NDS ↑	mAP ↑	mATE ↓	mAOE ↓
0.5	0.434	0.342	0.741	0.383
1.0	**0.437**	**0.349**	**0.725**	0.392
2.0	0.433	0.344	0.728	**0.376**

Table 7: Quantitative analysis of the two-stage weights.

Training Convergence

To compare the convergence rate and final performance of SimMOD and the baseline methods, we train three methods under the same schedule and summarize the results in Fig. 3. One can find that the monocular method FCOS3D (Tian et al. 2019) converges much faster than the top-down method DETR3D (Wang et al. 2021b), while DETR3D (Wang et al. 2021b) has a higher performance limit at convergence. By contrast, our proposed two-stage paradigm can possess all their advantages. On the one hand, SimMOD can benefit from the guidance in the image views and quickly learn to recognize foreground objects, contributing to faster convergence. On the other hand, the second stage can further utilize multi-scale and multi-view features to refine the proposals, pushing the performance limit to a higher level.

Ablation Studies

For efficiency, we use 1/4 of the training set for ablation experiments following (Chen et al. 2022b). We validate our designs in the following three aspects, including learning sample-wise proposals, promoting consistent supervision, and designing the proposal encodings.

Sample-wise proposals. Since our core contribution is the two-stage propose-and-fuse framework, the design choices of proposal generation play a key role in improving the overall detection performance. As shown in Tab. 4, we first introduce the prediction of objectness map and depths to generate the proposals. Compared with the fixed queries, the minimal design with proposals can already improve NDS from 36.2% to 40.1%. Then, a simple 3×3 max-pooling is conducted on the predicted objectness map to effectively remove duplicates and improve the performance by 0.8% NDS. Finally, the auxiliary branches are applied to enhance the feature learning, which brings a significant boost of 2.8% NDS.

Consistent supervision. In Tab. 5, we verify the effectiveness of target filtering and teacher forcing in guaranteeing consistent supervision. We can observe that both techniques can mitigate inconsistency and improve performance. When jointly applied, the consistent supervision can outperform the default baseline by 1.0% NDS.

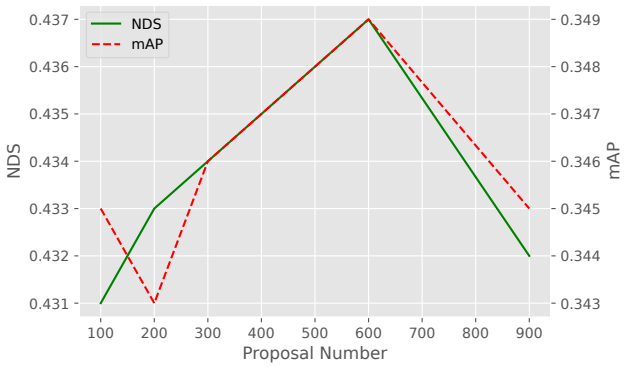

Figure 4: Qualitative analysis on the number of proposals. Though the number of proposals varies widely, the performance of SimMOD is rather stable.

Proposal encoding. Considering the proposal encoding, we validate the importance of incorporating the views and levels. As shown in Tab. 6, embedding the information of which camera and which level into the proposal can notably improve the overall performance. Since one object can potentially generate proposals at multiple views and levels, incorporating such information can help in removing duplicates and promoting inter-view interactions.

Hyperparamter Sensitivity

In this section, we investigate the influence of main hyperparameters, including the number of proposals and the weights for balancing the two-stage learning. Also, we use 1/4 of the training set for experiments.

Number of proposals. In Fig. 4, we show the performance of SimMOD with respect to different numbers of proposals. Thanks to the design of sample-wise proposals, SimMOD is relatively robust to the number of proposals. With only 100 proposals, SimMOD scores 43.1% NDS, which is slightly worse than the 43.7% NDS with 600 proposals. Besides, further increasing the number of proposals cannot improve the performance.

Two-stage loss weights. In Tab. 7, we analyze the influence of tuning the weight λ to balance the two-stage learning. Despite being a two-stage paradigm, SimMOD can achieve satisfactory performance with various values of λ.

Conclusion

In this paper, we present SimMOD: a simple two-stage propose-and-fuse framework for 3D object detection from multiple surrounding images. In the first stage, existing monocular detectors can be utilized to process single-view images and generate high-quality proposals. For the second stage, multi-view proposals are aggregated into the same 3D space and iteratively refined with attention mechanism and geometry-based feature sampling. Experimental results on the nuScenes detection benchmark demonstrate the superiority of SimMOD with new state-of-the-art performance. We hope the proposed framework can serve as a simple and strong baseline for multi-camera 3D object detection.

References

Bengio, S.; Vinyals, O.; Jaitly, N.; and Shazeer, N. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28.

Caesar, H.; Bankiti, V.; Lang, A. H.; Vora, S.; Liong, V. E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; and Beijbom, O. 2020. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*.

Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; and Zagoruyko, S. 2020. End-to-End Object Detection with Transformers. In *ECCV*.

Chen, P.; Liu, S.; Zhao, H.; and Jia, J. 2020. Gridmask data augmentation. *arXiv preprint arXiv:2001.04086*.

Chen, S.; Wang, X.; Cheng, T.; Zhang, Q.; Huang, C.; and Liu, W. 2022a. Polar parametrization for vision-based surround-view 3d detection. *arXiv preprint arXiv:2206.10965*.

Chen, X.; Kundu, K.; Zhang, Z.; Ma, H.; Fidler, S.; and Urtasun, R. 2016. Monocular 3d object detection for autonomous driving. In *CVPR*.

Chen, Z.; Li, Z.; Zhang, S.; Fang, L.; Jiang, Q.; and Zhao, F. 2022b. Graph-DETR3D: Rethinking Overlapping Regions for Multi-View 3D Object Detection. *arXiv preprint arXiv:2204.11582*.

Dai, J.; Qi, H.; Xiong, Y.; Li, Y.; Zhang, G.; Hu, H.; and Wei, Y. 2017. Deformable convolutional networks. In *ICCV*.

Ding, M.; Huo, Y.; Yi, H.; Wang, Z.; Shi, J.; Lu, Z.; and Luo, P. 2020. Learning depth-guided convolutions for monocular 3d object detection. In *CVPR*.

Girshick, R.; Donahue, J.; Darrell, T.; and Malik, J. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Huang, J.; Huang, G.; Zhu, Z.; and Du, D. 2021. Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. *arXiv preprint arXiv:2112.11790*.

Lang, A. H.; Vora, S.; Caesar, H.; Zhou, L.; Yang, J.; and Beijbom, O. 2019. Pointpillars: Fast encoders for object detection from point clouds. In *CVPR*.

Li, Z.; Wang, W.; Li, H.; Xie, E.; Sima, C.; Lu, T.; Yu, Q.; and Dai, J. 2022. BEVFormer: Learning Bird's-Eye-View Representation from Multi-Camera Images via Spatiotemporal Transformers. *arXiv preprint arXiv:2203.17270*.

Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; and Belongie, S. 2017a. Feature Pyramid Networks for Object Detection. In *CVPR*.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollar, P. 2017b. Focal Loss for Dense Object Detection. In *ICCV*.

Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; and Berg, A. C. 2016. SSD: Single Shot MultiBox Detector. In *ECCV*.

Liu, Y.; Wang, T.; Zhang, X.; and Sun, J. 2022. Petr: Position embedding transformation for multi-view 3d object detection. *arXiv preprint arXiv:2203.05625*.

Loshchilov, I.; and Hutter, F. 2019. Decoupled Weight Decay Regularization. In *ICLR*.

Lu, J.; Zhou, Z.; Zhu, X.; Xu, H.; and Zhang, L. 2022. Learning Ego 3D Representation as Ray Tracing. *arXiv preprint arXiv:2206.04042*.

Ma, X.; Zhang, Y.; Xu, D.; Zhou, D.; Yi, S.; Li, H.; and Ouyang, W. 2021. Delving into Localization Errors for Monocular 3D Object Detection. In *CVPR*.

Mousavian, A.; Anguelov, D.; Flynn, J.; and Kosecka, J. 2017. 3d bounding box estimation using deep learning and geometry. In *CVPR*.

Qin, Z.; Wang, J.; and Lu, Y. 2019. Triangulation learning network: from monocular to stereo 3d object detection. In *CVPR*.

Reading, C.; Harakeh, A.; Chae, J.; and Waslander, S. L. 2021. Categorical depth distribution network for monocular 3d object detection. In *CVPR*.

Redmon, J.; Divvala, S. K.; Girshick, R. B.; and Farhadi, A. 2016. You Only Look Once: Unified, Real-Time Object Detection. In *CVPR*.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *NeurIPS*.

Shi, S.; Wang, X.; and Li, H. 2019. Pointrcnn: 3d object proposal generation and detection from point cloud. In *CVPR*.

Sun, J.; Chen, L.; Xie, Y.; Zhang, S.; Jiang, Q.; Zhou, X.; and Bao, H. 2020a. Disp r-cnn: Stereo 3d object detection via shape prior guided instance disparity estimation. In *CVPR*.

Sun, P.; Kretzschmar, H.; Dotiwalla, X.; Chouard, A.; Patnaik, V.; Tsui, P.; Guo, J.; Zhou, Y.; Chai, Y.; Caine, B.; et al. 2020b. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*.

Sun, Z.; Cao, S.; Yang, Y.; and Kitani, K. M. 2021. Rethinking transformer-based set prediction for object detection. In *ICCV*.

Tian, Z.; Shen, C.; Chen, H.; and He, T. 2019. Fcos: Fully convolutional one-stage object detection. In *CVPR*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *NeurIPS*.

Wang, T.; Xinge, Z.; Pang, J.; and Lin, D. 2022. Probabilistic and geometric depth: Detecting objects in perspective. In *CoRL*.

Wang, T.; Zhu, X.; Pang, J.; and Lin, D. 2021a. Fcos3d: Fully convolutional one-stage monocular 3d object detection. In *ICCV*.

Wang, Y.; Chao, W.-L.; Garg, D.; Hariharan, B.; Campbell, M.; and Weinberger, K. Q. 2019. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *CVPR*.

Wang, Y.; Guizilini, V.; Zhang, T.; Wang, Y.; Zhao, H.; and Solomon, J. M. 2021b. DETR3D: 3D Object Detection from Multi-view Images via 3D-to-2D Queries. In *CoRL*.

Yu, F.; Wang, D.; Shelhamer, E.; and Darrell, T. 2018. Deep layer aggregation. In *CVPR*.

Zhang, Y.; Lu, J.; and Zhou, J. 2021. Objects are Different: Flexible Monocular 3D Object Detection. In *CVPR*.

Zhang, Y.; Zhu, Z.; Zheng, W.; Huang, J.; Huang, G.; Zhou, J.; and Lu, J. 2022. BEVerse: Unified Perception and Prediction in Birds-Eye-View for Vision-Centric Autonomous Driving. *arXiv preprint arXiv:2205.09743*.

Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; and Dai, J. 2021. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In *ICLR*.