Visual Point Cloud Forecasting enables Scalable Autonomous Driving

Zetong Yang Li Chen Yanan Sun Hongyang Li

OpenDriveLab and Shanghai AI Lab

https://github.com/OpenDriveLab/ViDAR

In memoriam Prof. Xiao'ou Tang

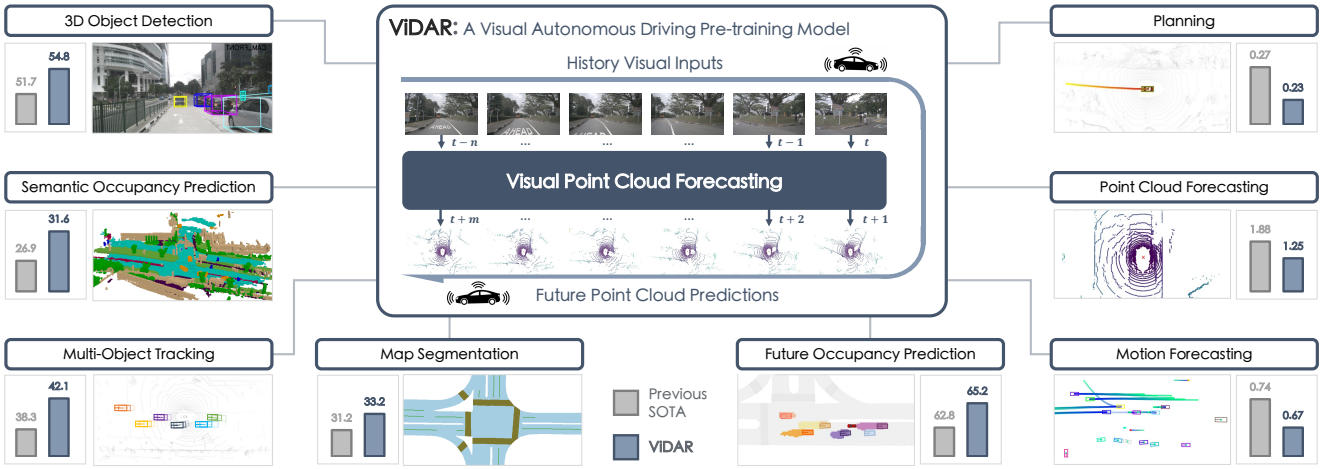

Figure 1. **ViDAR** is a visual autonomous driving pre-training framework, which leverages the estimation of future point clouds from historical visual inputs as the pre-text task. We term this new pre-text task as *visual point cloud forecasting*. With the aid of ViDAR, we achieve substantial improvement spanning a diverse spectrum of downstream applications for perception, prediction, and planning.

Abstract

In contrast to extensive studies on general vision, pre-training for scalable visual autonomous driving remains seldom explored. Visual autonomous driving applications require features encompassing semantics, 3D geometry, and temporal information simultaneously for joint perception, prediction, and planning, posing dramatic challenges for pre-training. To resolve this, we bring up a new pre-training task termed as visual point cloud forecasting - predicting future point clouds from historical visual input. The key merit of this task captures the synergic learning of semantics, 3D structures, and temporal dynamics. Hence it shows superiority in various downstream tasks. To cope with this new problem, we present ***ViDAR****, a general model to pre-train downstream visual encoders. It first extracts historical embeddings by the encoder. These representations are then transformed to 3D geometric space via a novel Latent Rendering operator for future point cloud prediction. Experiments show significant gain in downstream tasks, e.g., 3.1% NDS on 3D detection, ∼10% error reduction on motion forecasting, and ∼15% less collision rate on planning.*

1. Introduction

Recently, the community has witnessed rapid development in visual, or camera-only autonomous driving, with input being monocular or multi-view images [52, 64, 70, 71, 79]. Leveraging visual inputs alone, existing approaches demonstrate superior capability of extracting Bird's-Eye-View (BEV) features [6, 29, 47, 53, 62, 85], and performing well in perception [30, 43, 55, 68, 76], prediction [17, 28, 66], and planning [31, 32]. Despite significant improvements in applications, these models rely on precise 3D annotations to a great extent, which are often difficult to collect, *e.g.*, semantic occupancy [3, 10], 3D boxes [5, 16, 65], trajectories [15], and thus are challenging to scale up for production.

Considering the expensive labeling workflow, pre-training [2, 18, 63] has emerged as a crucial approach to scale up downstream applications. The key idea is to define

pretext tasks that leverage large amounts of readily available data to learn meaningful representations. This enhances downstream performance though labeled data is limited.

Though extensive research of pre-training in computer vision has been conducted [8, 20–22, 34, 67, 81, 82], its application in visual autonomous driving remains seldom explored. Visual autonomous driving poses great challenges in pre-training as it requires the features to maintain semantics, 3D geometry, and temporal dynamics at the same time for joint perception, prediction, and planning [7, 80]. As a result, most models still rely on supervised pre-training, like 3D detection [61, 70] or occupancy [59, 68, 84], using labeled data that is often unavailable at scale [40]. Some approaches propose estimating depth [61] or rendering masked scenes [86] as pre-training. They use Image-LiDAR pairs as a means of achieving scalable annotation-free pre-training. However, they struggle in either multi-view 3D geometry or temporal modeling (Figure 2). Depth estimation retrieves depth from one image, limited in multi-view geometry; rendering techniques reconstruct scenes from multi-view images but lacking temporal modeling. However, temporal modeling is crucial in end-to-end autonomous driving systems, *e.g.*, UniAD [28], especially for prediction and planning which are the ultimate goals and require accurate scene flow and object motion for decision-making. Due to the absence of temporal modeling, existing approaches are insufficient for pre-training the end-to-end system.

In this work, we explore pre-training tailored for end-to-end visual autonomous driving applications, including not only perception but also prediction and planning [7]. We formulate a new pre-text task, visual point cloud forecasting (Figure 2), to fully exploit information of semantics, 3D geometry, and temporal dynamics behind the raw Image-LiDAR sequences, with being scalable into consideration. It predicts future point clouds from historical visual images.

The main rationale of visual point cloud forecasting lies in the simultaneous supervision of semantics, 3D structure, and temporal modeling. By compelling the model to predict the future from history, it supervises the extraction of scene flow and object motion which are crucial for temporal modeling and future estimation. Meanwhile, it involves the reconstruction of point clouds from images, which supervises the multi-view geometry and semantic modeling. Therefore, features from visual point cloud forecasting embed information of both geometric and temporal hints, beneficial for perception, tracking, and planning simultaneously.

To this end, we present **ViDAR**, a general visual point cloud forecasting approach for pre-training (Figure 2). ViDAR includes three parts, *History Encoder*, *Latent Rendering* operator, and *Future Decoder*. The History Encoder is the target structure for pre-training. It could be any visual BEV encoder [47] to embed visual sequences into BEV

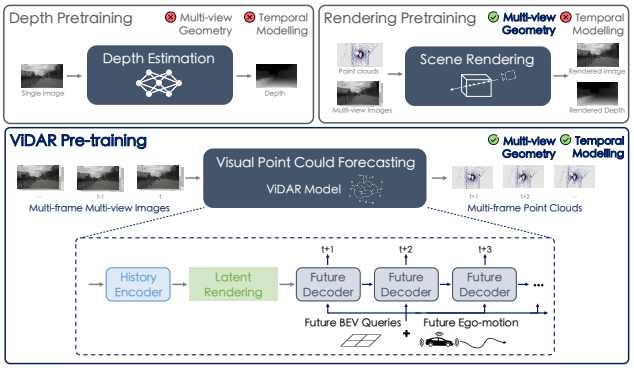

Figure 2. **Comparisons among visual autonomous driving pre-training paradigms and our ViDAR architecture.** Compared to existing methods, visual point cloud forecasting jointly models multi-view geometry and temporal dynamics. We then propose ViDAR, using Image-LiDAR sequences to pre-train visual encoders.

space. These BEV features are sent to the Latent Rendering operator. Latent Rendering plays a crucial role in enabling ViDAR benefit downstream performance. It solves the ray-shaped BEV features issue [48, 87], models 3D geometric latent space, and bridges encoder and decoder. The Future Decoder is an auto-regressive transformer that takes historical BEV features to iteratively predict future point clouds for arbitrary timestamps.

ViDAR provides a comprehensive solution for visual autonomous driving pre-training. We test ViDAR on nuScenes dataset [5] in terms of point cloud forecasting and downstream verifications. Though with visual inputs, ViDAR still outperforms previous forecasting methods using point clouds, ∼33% Chamfer Distance reduction on point cloud estimation of 1s future. ViDAR also improves downstream performance. Using Image-LiDAR sequences only, ViDAR surprisingly outperforms 3D detection pre-training [70], *e.g.*, by 1.1% mAP and 2.5% mIoU for detection and semantic occupancy prediction, under the same data scale. If also based on the 3D detection pre-training, ViDAR boosts previous methods by 4.3% mAP and 4.6% mIoU. Further, due to the effective pre-training on the joint capture of geometric and temporal information, ViDAR improves UniAD [28] on all tasks for end-to-end autonomous driving including perception, prediction, and planning by a large margin (Figure 1). Experimental results validate that visual point cloud forecasting enables scalable autonomous driving.

2. Related Work

Pre-training for Visual Autonomous Driving. Pre-training for scalable applications has been extensively studied in general vision. These approaches can be roughly divided into contrastive approaches [8, 20, 35, 67], which learn discriminative features from positive and negative pairs; and masked signal modeling [13, 21, 75, 83], which

recover discarded signals from remained signals to capture a comprehensive understanding of global semantics.

In contrast, pre-training for visual autonomous driving is still under-explored. Visual autonomous driving poses great challenges as it requires semantic understanding, 3D structural awareness, and temporal modeling at the same time, for joint perception, prediction, and planning. Existing vision methods mainly consider semantics; methods based on Image-LiDAR pairs [61, 86] struggle with temporal modeling; other supervised strategies [68, 70] are not scalable. In this work, we propose visual point cloud forecasting, which simultaneously models semantics, temporal dynamics, and 3D geometry by a uniform process, and is easily scaled up.

Point Cloud Forecasting. Point cloud forecasting, one of the most fundamental self-supervised tasks for autonomous driving, predicts future point clouds from past point cloud inputs. Previous works use range image [4, 58], a representation obtained by projecting point clouds to dense 2D images using sensor intrinsic and extrinsic parameters. Based on historical range images, they apply 3D convolutions [57], or LSTMs [77, 78] to predict future point clouds. Yet, they additionally model the motion of sensor intrinsic and extrinsic parameters. Later methods factor out the estimation of sensors by introducing 4D occupancy prediction [37] and differentiable ray-casting [36], which ensures a better modeling of the world. Compared to prior literature, we aim at visual point cloud forecasting, using past images to predict future point clouds. Meanwhile, we raise this task as a pre-training paradigm for visual autonomous driving and demonstrate its superiority in a wide range of downstream applications.

3. Methodology

In this section, we elaborate on our ViDAR, a visual point cloud forecasting approach for general autonomous driving pre-training. We begin with an overview of ViDAR in Section 3.1, and subsequently delve into Latent Rendering and Future Decoder in Section 3.2 and Section 3.3, respectively.

3.1. Overview

As depicted in Figure 2, ViDAR comprises three components: **(a)** a *History Encoder*, also the target structure of pre-training, which extracts BEV embeddings $\mathcal{F}_{\text{bev}}$ from visual sequence inputs $\mathcal{I}$; It can be any visual BEV encoder [29, 47, 53]; **(b)** a *Latent Rendering* operator, which simulates the volume rendering operation in latent space so as to obtain geometric embedding $\hat{\mathcal{F}}_{\text{bev}}$ from $\mathcal{F}_{\text{bev}}$; and **(c)** a *Future Decoder*, which predicts future BEV features $\hat{\mathcal{F}}_t$ at timestamps $t \in \{1, 2, ...\}$ in an auto-regressive manner. Finally, a prediction head is followed to project $\hat{\mathcal{F}}_t$ into 3D occupancy volume $\mathcal{P}_t$. This process is formulated as:

$$\begin{aligned} \mathcal{F}_{\text{bev}} &= \texttt{Encoder}(\mathcal{I}), \\ \hat{\mathcal{F}}_{\text{bev}} &= \texttt{LatentRender}(\mathcal{F}_{\text{bev}}), \\ \hat{\mathcal{F}}_t &= \texttt{Decoder}(\hat{\mathcal{F}}_{t-1}), \text{ where } \hat{\mathcal{F}}_0 = \hat{\mathcal{F}}_{\text{bev}}, \\ \mathcal{P}_t &= \texttt{Projection}(\hat{\mathcal{F}}_t). \end{aligned} \tag{1}$$

Point cloud predictions are obtained from the predicted occupancy volume $\mathcal{P}_t$. This process is similar to the previous point cloud forecasting method [37]. Specifically, we first cast rays from the origin to various designated directions, then figure out the distance of waypoints along each ray with the maximum occupancy response, and finally compute the point position according to the distance and corresponding ray direction.

3.2. Latent Rendering

A straightforward solution of visual point cloud forecasting for pre-training is to incorporate the History Encoder and Future Decoder directly with differentiable ray-casting [37], which is the crucial component in state-of-the-art point cloud forecasting methods to render point clouds from predicted occupancy volume and compute loss for backpropagation. However, our experimental results show that this approach does not yield improvements and even has a detrimental effect on the downstream tasks due to the defective geometric feature modeling ability.

Preliminary. Differentiable ray-casting is a volume rendering process operating on an occupancy volume, denoted as $\mathcal{P} \in \mathbb{R}^{L \times H \times W}$. It renders depths of various rays and subsequently converts the depths with corresponding ray directions to point clouds.

Formally, starting from the origin sensor position, $\mathbf{o} \in \mathbb{R}^3$, differentiable ray-casting casts n rays with varying directions, $\mathbf{d} \in \mathbb{R}^{n \times 3}$. Along each ray i, it uniformly samples m waypoints at different distances $\lambda^{(j)} \in \mathbb{R}, j \in \{1, 2, ..., m\}$ until reaching the boundary of the 3D space. The coordinates of these waypoints are calculated as:

$$\mathbf{x}^{(i,j)} = \mathbf{o} + \lambda^{(j)} \mathbf{d}^{(i)}. \tag{2}$$

Those waypoint coordinates, $\mathbf{x} \in \mathbb{R}^{n \times m \times 3}$, are used to compute occupancy values. This process is quantized [37], wherein the waypoints are discretized to occupancy volume grids. Then, the values of waypoints are derived as the associated values of volume grids, $\mathbf{p}^{(i,j)} = \mathcal{P}^{([\mathbf{x}^{(i,j)}])}$. Here $[\cdot]$ denotes a rounding operation for discretizing waypoints.

Differentiable ray-casting renders the corresponding depth of the i-th ray, $\hat{\lambda}^{(i)}$, by an integral process:

$$\hat{\mathbf{p}}^{(i,j)} = \left[\prod_{k=1}^{j-1} (1 - \mathbf{p}^{(i,k)}) \right] \mathbf{p}^{(i,j)}, \tag{3}$$

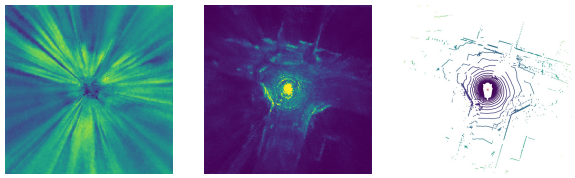

Figure 3. **Ray-shaped Features *vs*. Geometric Features.** Ray-shaped features show similar feature responses on BEV grids along the same ray; while geometric features from the Latent Rendering maintain discriminative 3D geometry and can describe the 3D world in latent space.

Forecasting Structure	N/A	Differentiable Ray-casting	Latent Rendering
NDS (%)	44.11	40.20 (**-3.91**)	47.58 (**+3.47**)

Table 1. **Downstream detection performance under different forecasting structures.** "N/A" represents the baseline without forecasting pre-training. We observe a performance drop when directly using History Encoder and Future Decoder with differentiable ray-casting for pre-training; while, with the Latent Rendering operator, the performance is significantly improved.

$$\hat{\lambda}^{(i)} = \sum_{j=1}^{m} \hat{\mathbf{p}}^{(i,j)} \lambda^{(j)}. \tag{4}$$

For simplicity, we name Eq. 3 and Eq. 4 as *conditional probability function* and *distance expectation function*. The conditional probability function determines the occupancy of a grid by considering the conditional probability that prior waypoints are unoccupied and the ray terminates at this particular grid; the distance expectation function retrieves depths from the occupancy of grids in 3D volume. L1 loss is then applied to supervise the rendered depth for training point cloud forecasting.

Despite the great success of differentiable ray-casting in the task of point cloud forecasting, its application in visual point cloud forecasting pre-training does not bring any benefit for downstream performance (Table 1). After such pre-training, ray-shaped features [48, 87], where grids along the same ray tend to possess similar features (Figure 3 - (a.)), are observed. The underlying reason is that waypoints along the same ray in 3D space usually correspond to the same pixel in the visual image, resulting in a tendency to learn similar feature responses. As a consequence, these ray-shaped features are not discriminative and representative enough when transferred to downstream applications, leading to reduced performance.

Latent Rendering. In order to extract more discriminative and representative features, we introduce the Latent Rendering operator. It first computes the ray-wise feature through a *feature expectation function*, then customizes features of each grid by weighting the ray-wise feature with its

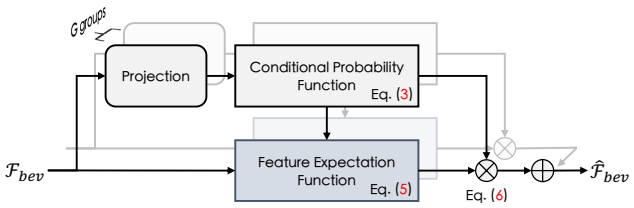

Figure 4. **Multi-group Latent Rendering** comprises several Latent Rendering running in parallel for different channels. Latent Rendering captures geometric features by the conditional probability function and the feature extraction function. "$\bigoplus$" means concatenating multi-group features among channel dimensions.

associated conditional probability. The overall structure is depicted in Figure 4.

To be specific, inspired by Eq. (4), the feature expectation function is formulated in a similar form:

$$\hat{\mathcal{F}}^{(i)} = \sum_{k=1}^{m} \hat{\mathbf{p}}^{(i,k)} \mathcal{F}_{\text{bev}}^{(k)}, \tag{5}$$

where i represents the ray extending from the origin to the i-th BEV grid. Here, $\hat{\mathbf{p}}$ is the conditional probability, computed through the conditional probability function Eq. (3), which takes as input the learnable independent probability projected from $\mathcal{F}_{bev}$. The ray-wise features are shared by all grids lying in the same ray.

Then, we compute the grid feature as:

$$\hat{\mathcal{F}}_{\text{bev}} = \hat{\mathbf{p}} \cdot \hat{\mathcal{F}}, \tag{6}$$

which highlights the response of BEV grids with higher conditional probability so as to make $\hat{\mathcal{F}}_{bev}$ discriminative. This enables the BEV encoder to learn the geometric features during pre-training (Figure 3 - (b.)).

To enhance the diversity of geometric features, we further design the multi-group Latent Rendering. By parallelizing multiple Latent Rendering on different feature channels, we allow ray-wise features to maintain diverse information, leading to better downstream performance.

As described in Eq. (3), the conditional probability of each BEV grid is determined not only by its own independent response, but also by the response of all its prior grids. Consequently, in the pre-training phase, once the model raises the response of a particular BEV grid, the corresponding responses of all its prior and subsequent grids are suppressed, which mitigates the issue of ray-shaped features during pre-training. After the pre-training with Latent Rendering, it is generally observed that there are only a few peaks with higher responses on a specific ray, indicating the presence of objects or structures in the scene. This effectively promotes a more accurate and consistent understanding of the 3D environment.

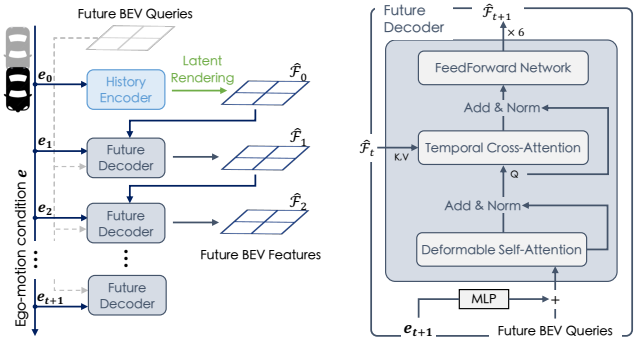

Figure 5. **Future Decoder** iteratively predicts the next BEV features, $\hat{\mathcal{F}}_t$, from the conditions of ego-motion $\mathbf{e}_t$ and the last BEV features, to enable specific future predictions with any ego-control.

3.3. Future Decoder

The Future Decoder predicts the next BEV features $\hat{\mathcal{F}}_t$ of frame t based on the inputs of previous BEV latent space $\hat{\mathcal{F}}_{t-1}$ and the expected next ego-motion, $\mathbf{e}_t$. The predicted features are then used to generate point clouds as Eq. (1).

Architecture. As depicted in Figure 5, Future Decoder is a transformer that can be iteratively used to predict future BEV features from the last roll-out embeddings in an auto-regressive manner. In the t-th iteration, it first encodes the ego-motion condition $\mathbf{e}_t$, which describes the expected coordinates and heading of ego-vehicle in the next frame, into high-dimensional embeddings by multi-layer perceptron (MLP), which are then added to future BEV queries as inputs of the transformer. Then, 6 transformer layers, composed of a Deformable Self-Attention [91], a Temporal Cross-Attention and a FeedForward Network [14], are used to predict the future $\hat{\mathcal{F}}_t$ based on the condition and the last BEV features $\hat{\mathcal{F}}_{t-1}$.

The Temporal Cross-Attention layer follows the design of Deformable Cross-Attention [91]. The difference lies in the reference coordinates of query points. In the context of Deformable Cross-Attention [91], "reference coordinates" refer to the corresponding positions of query points on the feature maps of keys and values. Typically, they are consistent. However, as for Future Decoder, due to the moving of the ego-vehicle, the ego-coordinate systems between the last and target frame are not necessarily aligned. Therefore, we additionally compute the reference coordinates of future BEV queries in previous BEV feature maps, according to the ego-motion condition, to align coordinate systems.

After obtaining the next BEV features $\hat{\mathcal{F}}_t$, we use a projection layer to generate the occupancy volume $\mathcal{P}_t$.

Loss. Instead of using L1 loss to supervise the depths of various rays, we directly apply ray-wise cross-entropy loss to maximize the response of points along its corresponding ray, as we have already obtained geometric features after the Latent Rendering operator. To be specific, for each ground-truth point of the t-th future point clouds, we cast a ray from the origin position $\mathbf{o}$ (the sensor position) towards the point, uniformly sample some waypoints along the ray until out of the volume, and compute cross-entropy loss for the ray to maximize the response of the point position and minimize the response of other waypoint positions. This process is formulated as:

$$\mathcal{L} = -\frac{1}{Tn}\sum_{t=1}^{T}\sum_{i=1}^{n}\log\left(\frac{e^{\mathcal{P}_t^{(\mathbf{g}^{(i)})}}}{\sum_j e^{\mathcal{P}_t^{(\mathbf{x}^{(i,j)})}} + e^{\mathcal{P}_t^{(\mathbf{g}^{(i)})}}}\right), \quad (7)$$

where T, n indicate the number of future supervisions and the number of points in the t-th ground-truth point clouds. $\mathbf{g}^{(i)}$ and $\mathbf{x}^{(i,j)}$ are the coordinates of i-th ground-truth point and j-th waypoints along the same ray. $\mathcal{P}_t^{(\cdot)}$ is trilinear interpolation to obtain corresponding values from volume $\mathcal{P}_t$.

4. Experiments

This section investigates the following questions:

- Can future point clouds be estimated from visual history, and how about ViDAR compared to point cloud methods?
- Can ViDAR help perception, prediction, and planning at the same time so as for scalable autonomous driving?
- Can ViDAR reduce the reliance of downstream applications on precise human annotations?
- How do different modules affect final performance?

4.1. Setup

Dataset. We conduct experiments on the challenging nuScenes [5] dataset, which is a large-scale dataset with 1,000 autonomous driving sequences. This dataset is widely utilized in perception tasks including 3D object detection [29, 45–47], multi-object tracking [25, 60, 89], and semantic occupancy prediction [10, 68]. It has also become a popular benchmark for subsequent research on end-to-end autonomous driving, including map segmentation [41, 69], trajectory predictions [17, 49], future occupancy prediction [23, 90], and open-loop planning [27, 28, 33].

Implementation Details. We base our implementation on mmDet3D codebase [9] and conduct downstream verifications on BEVFormer for 3D detection, OccNet [68] for semantic occupancy prediction, and UniAD [28] for unified perception, prediction, and planning. We choose those downstream baselines due to their effectiveness on a wide range of tasks and sharing the same BEV encoder structure, BEVFormer encoder [47]. Without any specifications, the default historical encoder of ViDAR is the BEVFormer-base encoder, consisting of a ResNet101-DCN [11, 19] backbone with an FPN neck [51] and additional 6 encoder layers to extract BEV features from multi-view image sequences, which is consistent with downstream models.

To render geometric features, we use a 16-group Latent Rendering. Each group is responsible for rendering latent spaces of 16 channels given the features of 256 channels after the BEVFormer-base encoder. The Future Decoder is a 6-layer structure with a channel of 256 for each. The future BEV queries are 200×200 learnable tokens indicating a valid perception range of [-51.2m, 51.2m] for the X and Y axis. We then use a projection layer with output channels as 16 to transform the predicted future BEV features to occupancy volume prediction $\mathcal{P} \in \mathbb{R}^{200\times200\times16}$, where 16 indicates the height dimension with a range of [-5m, 3m].

During pre-training, we use 5 frames of historical multi-view images and iterate the Future Decoder 6 times to predict point clouds for the future 3 seconds (each frame has 0.5 second interval). In each training step, we randomly select 1 future prediction for computing loss and detach the gradients of the others to save GPU memory. We pre-train the system for 50 epochs by AdamW optimizer [38, 56] with an initial learning of 2e-4 adjusted by cosine annealing strategy. For fine-tuning, we follow the same training strategy as the officially released downstream models.

4.2. Main Results

We now demonstrate the effectiveness of ViDAR across different tasks. First, we test the ability of ViDAR as a point cloud forecasting framework and compare it with the state-of-the-art approach that uses LiDAR inputs. Then, we show its advancement as a visual autonomous driving pre-training solution. We report the downstream comparison results in the order of perception-prediction-planning with previous state-of-the-art models on the nuScenes validation dataset.

Downstream Settings. For downstream verifications, we test ViDAR to pre-train BEV encoders under different initialization settings, listed as the following:

- **ViDAR-cls:** The BEV encoders are initialized with the backbone pre-trained for ImageNet classification [12], followed by ViDAR pre-training on nuScenes dataset.
- **ViDAR-2D-det:** The BEV encoder backbones are first pre-trained for 2D object detection on COCO dataset [50] before ViDAR pre-training on nuScenes dataset.
- **ViDAR-3D-det:** The BEV encoder backbones are pre-trained first by 3D detection on the nuScenes dataset, using FCOS3D [70], before ViDAR pre-training.

For UniAD experiments, after ViDAR pre-training, we first fine-tune BEVFormer for 3D detection, which is then used as the initialization for subsequent two-stage fine-tuning, consistent with the UniAD official implementation.

Point Cloud Forecasting. In Table 2, we present comparisons between our ViDAR and the previous state-of-the-art point cloud forecasting method, 4D-Occ [37]. The evaluation metric is Chamfer Distance. We evaluate both methods using input time horizons of 1s and 3s, corresponding to input sequences of 2 frames and 6 frames, respectively, following the same setup as in 4D-Occ. To provide detailed comparisons of performance, we report the quantitative forecasting results for each future timestamp. Only points within the range of [-51.2m, 51.2m] on the X- and Y-axis are considered during evaluation.

History Horizon	Method	Modality	Chamfer Distance (m^2) ↓ 0.5s	1.0s	1.5s	2.0s	2.5s	3.0s
1s	4D-Occ [37]	L	1.26	1.88	-	-	-	-
	ViDAR	C	**1.11**	**1.25**	**1.40**	**1.57**	**1.76**	**1.97**
3s	4D-Occ [37]	L	**0.91**	1.13	1.30	1.53	1.72	2.11
	ViDAR	C	1.01	**1.12**	**1.25**	**1.38**	**1.54**	**1.73**

Table 2. **Point cloud forecasting.** ViDAR surpasses prior state-of-the-art method on future prediction, using visual input only.

Methods	Encoder	Pre-train	mAP (%)	NDS (%)
BEV-Former [47]	RN50 [19]	ImageNet-cls [12]	25.2	35.4
		ViDAR-cls	**29.0**	**38.8**
	RN101 [19]	ImageNet-cls [12]	37.7	47.7
		ViDAR-cls	**42.6**	**51.8**
		nus-3D-det [70]	41.5	51.7
		ViDAR-3D-det	**45.8**	**54.8**
	Intern-S [72]	COCO-2D-det [50]	41.5	51.2
		ViDAR-2D-det	**47.6**	**56.4**
	Intern-B [72]	COCO-2D-det [50]	42.9	52.0
		ViDAR-2D-det	**50.1**	**57.6**

Table 3. **Detection performance** of BEVFormer with and without ViDAR pre-training under various backbones and initializations.

As presented in Table 2, ViDAR consistently outperforms 4D-Occ on both 1s and 3s settings, despite utilizing visual inputs exclusively. Specifically, with 1s history input, ViDAR achieves remarkable improvement over 4D-Occ, reducing forecasting errors by $\sim$33% for future 1s predictions. When using 3s inputs, we observe a $\sim$ 18% error reduction for 3s forecasting. Moreover, due to the auto-regressive design of our Future Decoder, ViDAR effectively predicts arbitrary future, though with the constraint of a limited 1s input horizon. These experiments demonstrate the effectiveness of ViDAR for point cloud forecasting.

Perception. We verify ViDAR on four downstream perception tasks, 3D object detection, semantic occupancy prediction, map segmentation, and multi-object tracking. In Table 3 and Table 4, we compare the performance of BEVFormer [47] and OccNet [68] with and without ViDAR pre-training under different backbones and initialization settings for 3D detection and semantic occupancy prediction, respectively. Notably, ViDAR, using solely Image-LiDAR sequences, outperforms 3D detection supervised pre-training (The 4th & the 5th row in Table 3, 42.6%

Methods	Encoder	Pre-train	mIoU (%) ↑
OccNet [68]	RN101 [19]	ImageNet-cls [12]	24.35
		ViDAR-cls	**29.57**
		nus-3D-det [70]	26.98
		ViDAR-3D-det	**31.67**
	Intern-S [72]	COCO-2D-det [50]	24.92
		ViDAR-2D-det	**30.51**
	Intern-B [72]	COCO-2D-det [50]	25.24
		ViDAR-2D-det	**31.69**

Table 4. **Semantic Occupancy Prediction.** ViDAR consistently improves OccNet on different backbones and initializations.

Methods	Encoder	Pre-train	Lanes (%) ↑
BEVFormer [47]	RN101	nus-3D-det [70]	23.9
UniAD [28]	RN101	nus-3D-det [47]	31.3
		ViDAR-3D-det	**33.2**

Table 5. **Map segmentation.** ViDAR improves UniAD on online mapping. The metric is segmentation IoU.

Methods	Encoder	Pre-train	AMOTA (%) ↑
ViP3D [17]	RN50	nus-3D-det [74]	21.7
QD3DT [25]	RN101	-	24.2
MUTR3D [89]	RN101	ImageNet-cls	29.4
DQTrack-DETR3D [46]	RN101	nus-3D-det [74]	36.7
DQTrack-UVTR [46]	RN101	nus-3D-det [43]	39.6
DQTrack-Stereo [46]	RN101	nus-3D-det [44]	40.7
DQTrack-PETRv2 [46]	V2-99	nus-3D-det [54]	44.6
UniAD-Stage1 [28]	RN101	nus-3D-det [47]	39.0
		ViDAR-3D-det	**45.1**

Table 6. **Multi-object tracking.** With ViDAR, UniAD outperforms previous end-to-end trackers using solely visual images.

mAP to 41.5% mAP; The 2nd & the 3rd row in Table 4, 29.57% mIoU to 26.98% mIoU). Furthermore, we also observe huge improvements in map segmentation (Table 5) and multi-object tracking (Table 6). These experiments demonstrate the effectiveness of ViDAR as a scalable pre-training method for enhancing 3D geometry modeling.

Prediction. The motion forecasting comparisons are presented in Table 7. As shown, ViDAR significantly improves the performance of UniAD [28]. For instance, we observe a ∼10% error reduction in minADE and a 3.5% improvement in EPA. In Table 8, we provide the comparison between UniAD with and without ViDAR pre-training in future occupancy prediction. The results demonstrate that ViDAR enhances the performance of UniAD for all areas. We observe improvements of 2.4% IoU and 2.7% VPQ for nearby areas, as well as improvements of 2.0% IoU and 2.5% VPQ for distant areas. With ViDAR pre-training, UniAD overcomes its limitation in occupancy forecasting for distant objects and now outperforms BEVerse [90] in all areas. These experiments highlight the effectiveness of ViDAR in enhancing downstream models in utilizing temporal information and improving their prediction performance.

Methods	minADE (m) ↓	minFDE (m) ↓	MR ↓	EPA ↑
PnPNet [49]	1.15	1.95	0.226	0.222
ViP3D [17]	2.05	2.84	0.246	0.226
UniAD [28]	0.75	1.08	0.158	0.463
ViDAR	**0.67**	**0.99**	**0.149**	**0.498**

Table 7. **Motion forecasting**. ViDAR effectively enhances temporal modeling, which in turn boosts UniAD in future motion forecasting, showing consistent improvements on different metrics.

Methods	VPQ-n. ↑	VPQ-f. ↑	IoU-n. ↑	IoU-f. ↑
Fiery [23]	50.2	29.9	59.4	36.7
StretchBEV [1]	46.0	29.0	55.5	37.1
ST-P3 [27]	-	32.1	-	38.9
BEVerse [90]	54.3	36.1	61.4	40.9
UniAD [28]	54.6	33.9	62.8	40.1
ViDAR	**57.3**	**36.4**	**65.4**	**42.1**

Table 8. **Future occupancy prediction.** ViDAR improves UniAD on future occupancy prediction for objects in both near (noted as "n.", 30x30m) and far (noted as "f.", 50x50m) evaluation areas.

Methods	Modality	Avg.Col. (3s) (%) ↓	Avg.L2 (3s) (m) ↓
FF [26]	L	0.43	1.43
EO [36]	L	0.33	1.60
ST-P3 [27]	C	0.71	2.11
VAD [33]	C	0.41	1.05
UniAD [28]	C	0.27	1.12
ViDAR	C	**0.23**	**0.91**

Table 9. **Planning.** ViDAR improves UniAD in terms of both collision avoidance and planning accuracy. Note that, the reported numbers are obtained by averaging the results of each timestamp in the future 3 seconds, which is consistent with the reported results of VAD [33] and ST-P3 [27] at the 3s timestamp instead of the averaged one. Please refer to GitHub:Issue for more details.

Planning. Due to the effective temporal modeling and advanced future prediction capabilities, ViDAR significantly improves UniAD by reducing its average collision rate within 3 seconds by ∼15%. Moreover, it achieves a substantial decrease in the average planning displacement error by 0.21m and enables UniAD to outperform the state-of-the-art method, VAD [33], on nuScenes open-loop planning evaluation. These improvements demonstrate the effectiveness of ViDAR as a valuable pre-training approach for end-to-end autonomous driving. The enhanced performance in collision avoidance and planning accuracy highlights the potential of ViDAR in enhancing the safety and efficiency of downstream autonomous driving applications.

Joint Perception-Prediction-Planning. Finally, we summarize the improvements of ViDAR on the state-of-the-art end-to-end visual autonomous driving system, UniAD [28], for joint perception, prediction, and planning. As de-

Method	Detection		Tracking			Mapping		Motion Forecasting			Future Occupancy Prediction				Planning	
	NDS ↑	mAP ↑	AMOTA↑	AMOTP↓	IDS↓	IoU-lane↑	IoU-road↑	minADE↓	minFDE↓	MR↓	IoU-n.↑	IoU-f.↑	VPQ-n.↑	VPQ-f.↑	avg.L2↓	avg.Col.↓
UniAD	49.36	37.96	38.3	1.32	1054	31.3	69.1	0.75	1.08	0.158	62.8	40.1	54.6	33.9	1.12	0.27
ViDAR	**52.57**	**42.33**	**42.0**	**1.25**	**991**	**33.2**	**71.4**	**0.67**	**0.99**	**0.149**	**65.4**	**42.1**	**57.3**	**36.4**	**0.91**	**0.23**

Table 10. **Performance gain of ViDAR for joint perception, prediction, and planning.** ViDAR consistently improves UniAD [28] on all tasks towards end-to-end autonomous driving, validating its effectiveness for scalable visual autonomous driving.

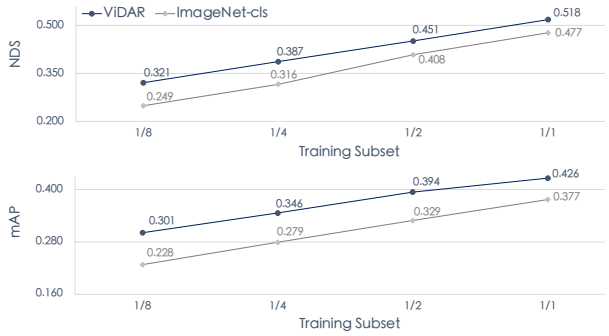

Figure 6. **Validation of ViDAR on Fine-tuning with limited supervised data.** We verify ViDAR on supervision efficiency by reducing available annotations for 3D object detection during downstream fine-tuning (from the full training set to a 1/8th subset) and observe a continuous improvement on each subset.

Groups of Latent R.	N/A	1	2	4	8	16
NDS (%)	40.20	39.18	43.36	45.53	47.01	**47.58**

Table 11. **Ablation of Latent Rendering for downstream fine-tuning.** We compare the performance of 3D detection pre-trained by ViDAR without the Latent Rendering operation (denoted as "N/A") and by ViDAR with different groups of Latent Rendering.

picted in Table 10, ViDAR brings substantial improvements on all sub-modules of UniAD for perception (Detection, Tracking, Mapping), prediction (Motion Forecasting and Future Occupancy Prediction), and planning at the same time. These consistent improvements illustrate that visual point cloud forecasting effectively exploits the information of semantics, 3D geometry, and temporal dynamics behind the easily obtainable Image-LiDAR sequences. This, consequently, enables scalable visual autonomous driving.

4.3. Ablative Study

We conduct further analysis of ViDAR on improving the performance of downstream models with limited supervised data, and the effect of the Latent Rendering operation on learning 3D geometric latent space. More ablation studies can be found in the supplementary materials.

Efficiency of Supervised Pre-training. The primary objective of pre-training is to minimize the dependence on precise 3D annotations. In Figure 6, we demonstrate the effectiveness of ViDAR in reducing the reliance of modern 3D detectors on accurate 3D box annotations. We fine-tune BEVFormer-base using partial 3D annotations on nuScenes, ranging from the full dataset to a $1/8$ subset.

As depicted in Figure 6, ViDAR exhibits a remarkable reduction in the dependence on 3D annotations. Notably, BEVFormer, pre-trained by ViDAR, surpasses its counterpart under full supervision by 1.7% mAP, while using only half of the supervised samples, *i.e.*, 39.4% mAP *vs*. 37.7% mAP. Consequently, through ViDAR, we can reduce half of 3D annotations without sacrificing precision. Additionally, we observe a consistent trend of increasing improvements as the available supervision decreases. For instance, the mAP improvements are 4.9%, 6.5%, 6.7%, and 7.3% when fine-tuned on the full, half, a quarter, and 1/8th subsets. These results highlight the potential of ViDAR in harnessing large amounts of Image-LiDAR sequences.

Effect of Latent Rendering operator. Latent Rendering is the key component of ViDAR, which enables visual point cloud forecasting to effectively contribute to downstream applications. It addresses the ray-shaped features issue encountered during pre-training. In Table 11, we verify its effectiveness by comparing the performance of downstream models pre-trained by ViDAR with the Latent Rendering or not. The downstream model is BEVFormer-small [47] for 3D object detection. For context, the baseline performance with ImageNet-cls pre-training is 44.11% NDS.

As depicted in Table 11, when the Latent Rendering is missing (denoted as "N/A"), also referred to as the baseline in Section 3.2, a significant decline is observed in downstream performance after fine-tuning, from 44.11% NDS to 40.20% NDS. In contrast, with the 16-group Latent Rendering, the performance improves to 47.58% NDS, a notable 3.47% NDS improvement over the baseline.

We conduct a comparison of Latent Rendering with different parallel groups in Table 11 as well. The results demonstrate a consistent improvement by dividing channels into more groups and integrating information separately.

5. Conclusion

In this paper, we introduced visual point cloud forecasting, which predicts future point clouds from historical visual images, as a new pre-training task for end-to-end autonomous driving. We developed ViDAR, a general model to pre-train visual BEV encoders, and designed a Latent Rendering operator to solve the ray-shaped feature issue. To conclude, our work demonstrates that visual point cloud forecasting

enables scalable autonomous driving.

Limitations and Future Work. Though with the potential of scalability, in this paper, we mainly conduct pre-training on Image-LiDAR sequences from nuScenes dataset, of which the data scale is still limited. As for the future, we plan to scale up the pre-training data of ViDAR, study visual point cloud forecasting across diverse datasets, and use publicly available Image-LiDAR sequences as much as possible to train a foundation visual autonomous driving model [39].

Acknowledgement

OpenDriveLab is the autonomous driving team affiliated with Shanghai AI Lab. This work was supported by National Key R&D Program of China (2022ZD0160104), NSFC (62206172), and Shanghai Committee of Science and Technology (23YF1462000). We thank team members from OpenDriveLab for valuable feedback along the project.

References

[1] Adil Kaan Akan and Fatma Güney. StretchBEV: Stretching Future Instance Prediction Spatially and Temporally. In *ECCV*, 2022. 7

[2] Randall Balestriero, Mark Ibrahim, Vlad Sobal, Ari Morcos, Shashank Shekhar, Tom Goldstein, Florian Bordes, Adrien Bardes, Gregoire Mialon, Yuandong Tian, Avi Schwarzschild, Andrew Gordon Wilson, Jonas Geiping, Quentin Garrido, Pierre Fernandez, Amir Bar, Hamed Pirsiavash, Yann LeCun, and Micah Goldblum. A Cookbook of Self-supervised Learning. *arXiv preprint arXiv:2304.12210*, 2023. 1

[3] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *ICCV*, 2019. 1

[4] Alex Bewley, Pei Sun, Thomas Mensink, Dragomir Anguelov, and Cristian Sminchisescu. Range Conditioned Dilated Convolutions for Scale Invariant 3D Object Detection. *arXiv preprint arXiv:2005.09927*, 2021. 3

[5] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A Multimodal Dataset for Autonomous Driving. In *CVPR*, 2020. 1, 2, 5

[6] Li Chen, Chonghao Sima, Yang Li, Zehan Zheng, Jiajie Xu, Xiangwei Geng, Hongyang Li, Conghui He, Jianping Shi, Yu Qiao, and Junchi Yan. PersFormer: 3D Lane Detection via Perspective Transformer and the OpenLane Benchmark. In *ECCV*, 2022. 1

[7] Li Chen, Penghao Wu, Kashyap Chitta, Bernhard Jaeger, Andreas Geiger, and Hongyang Li. End-to-end Autonomous Driving: Challenges and Frontiers. *arXiv preprint arXiv:2306.16927*, 2023. 2, 13

[8] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved Baselines with Momentum Contrastive Learning. *arXiv preprint arXiv:2003.04297*, 2020. 2

[9] MMDetection3D Contributors. MMDetection3D: OpenMMLab next-generation platform for general 3D object detection. https://github.com/open-mmlab/mmdetection3d, 2020. 5

[10] OpenScene Contributors. OpenScene: The Largest Up-to-Date 3D Occupancy Prediction Benchmark in Autonomous Driving, 2023. 1, 5

[11] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable Convolutional Networks. In *ICCV*, 2017. 5

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009. 6, 7

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*, 2018. 2

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ICLR*, 2021. 5

[15] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles Qi, Yin Zhou, Zoey Yang, Aurélien Chouard, Pei Sun, Jiquan Ngiam, Vijay Vasudevan, Alexander McCauley, Jonathon Shlens, and Dragomir Anguelov. Large Scale Interactive Motion Forecasting for Autonomous Driving: The Waymo Open Motion Dataset. In *ICCV*, 2021. 1

[16] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *I. J. Robotics Res.*, 2013. 1

[17] Junru Gu, Chenxu Hu, Tianyuan Zhang, Xuanyao Chen, Yilun Wang, Yue Wang, and Hang Zhao. ViP3D: End-to-end visual trajectory prediction via 3d agent queries. In *CVPR*, 2023. 1, 5, 7

[18] Jie Gui, Tuo Chen, Jing Zhang, Qiong Cao, Zhenan Sun, Hao Luo, and Dacheng Tao. A Survey on Self-supervised Learning: Algorithms, Applications, and Future Trends. *arXiv preprint arXiv:2301.05712*, 2023. 1

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016. 5, 6, 7

[20] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum Contrast for Unsupervised Visual Representation Learning. In *CVPR*, 2020. 2

[21] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked Autoencoders Are Scalable Vision Learners. In *CVPR*, 2022. 2

[22] Ji Hou, Benjamin Graham, Matthias Nießner, and Saining Xie. Exploring Data-Efficient 3D Scene Understanding with Contrastive Scene Contexts. In *CVPR*, 2021. 2

[23] Anthony Hu, Zak Murez, Nikhil Mohan, Sofía Dudas, Jeffrey Hawke, Vijay Badrinarayanan, Roberto Cipolla, and Alex Kendall. FIERY: Future Instance Segmentation in Bird's-Eye view from Surround Monocular Cameras. In *ICCV*, 2021. 5, 7

[24] Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. GAIA-1: A Generative World Model for Autonomous Driving. *arXiv preprint arXiv:2309.17080*, 2023. 13

[25] Hou-Ning Hu, Yung-Hsu Yang, Tobias Fischer, Trevor Darrell, Fisher Yu, and Min Sun. Monocular Quasi-Dense 3D Object Tracking. *TPAMI*, 2022. 5, 7

[26] Peiyun Hu, Aaron Huang, John Dolan, David Held, and Deva Ramanan. Safe Local Motion Planning with Self-Supervised Freespace Forecasting. In *CVPR*, 2021. 7

[27] Shengchao Hu, Li Chen, Penghao Wu, Hongyang Li, Junchi Yan, and Dacheng Tao. ST-P3: End-to-end Vision-based Autonomous Driving via Spatial-Temporal Feature Learning. In *ECCV*, 2022. 5, 7

[28] Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, Lewei Lu, Xiaosong Jia, Qiang Liu, Jifeng Dai, Yu Qiao, and Hongyang Li. Planning-oriented Autonomous Driving. In *CVPR*, 2023. 1, 2, 5, 7, 8, 13, 15

[29] Junjie Huang, Guan Huang, Zheng Zhu, Ye Yun, and Dalong Du. BEVDet: High-performance Multi-camera 3D Object Detection in Bird-Eye-View. *arXiv preprint arXiv:2112.11790*, 2021. 1, 3, 5

[30] Yuanhui Huang, Wenzhao Zheng, Yunpeng Zhang, Jie Zhou, and Jiwen Lu. Tri-Perspective View for Vision-Based 3D Semantic Occupancy Prediction. In *CVPR*, 2023. 1

[31] Xiaosong Jia, Yulu Gao, Li Chen, Junchi Yan, Patrick Langechuan Liu, and Hongyang Li. DriveAdapter: Breaking the Coupling Barrier of Perception and Planning in End-to-End Autonomous Driving. In *ICCV*, 2023. 1

[32] Xiaosong Jia, Penghao Wu, Li Chen, Jiangwei Xie, Conghui He, Junchi Yan, and Hongyang Li. Think Twice before Driving: Towards Scalable Decoders for End-to-End Autonomous Driving. In *CVPR*, 2023. 1

[33] Bo Jiang, Shaoyu Chen, Qing Xu, Bencheng Liao, Jiajie Chen, Helong Zhou, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. VAD: Vectorized Scene Representation for Efficient Autonomous Driving. In *ICCV*, 2023. 5, 7

[34] Li Jiang, Zetong Yang, Shaoshuai Shi, Vladislav Golyanik, Dengxin Dai, and Bernt Schiele. Self-supervised Pre-training with Masked Shape Prediction for 3D Scene Understanding. In *CVPR*, 2023. 2

[35] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised Contrastive Learning. *arXiv preprint arXiv:2004.11362*, 2020. 2

[36] Tarasha Khurana, Peiyun Hu, Achal Dave, Jason Ziglar, David Held, and Deva Ramanan. Differentiable Raycasting for Self-Supervised Occupancy Forecasting. In *ECCV*, 2022. 3, 7

[37] Tarasha Khurana, Peiyun Hu, David Held, and Deva Ramanan. Point Cloud Forecasting as a Proxy for 4D Occupancy Forecasting. In *CVPR*, 2023. 3, 6

[38] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *ICLR*, 2015. 6

[39] Hongyang Li, Yang Li, Huijie Wang, Jia Zeng, Pinlong Cai, Huilin Xu, Dahua Lin, Junchi Yan, Feng Xu, Lu Xiong, Jingdong Wang, Futang Zhu, Kai Yan, Chunjing Xu, Tiancai Wang, Beipeng Mu, Shaoqing Ren, Zhihui Peng, and Yu Qiao. Open-sourced Data Ecosystem in Autonomous Driving: the Present and Future. *arXiv preprint arXiv:2312.03408*, 2023. 9

[40] Hongyang Li, Chonghao Sima, Jifeng Dai, Wenhai Wang, Lewei Lu, Huijie Wang, Jia Zeng, Zhiqi Li, Jiazhi Yang, Hanming Deng, Hao Tian, Enze Xie, Jiangwei Xie, Li Chen, Tianyu Li, Yang Li, Yulu Gao, Xiaosong Jia, Si Liu, Jianping Shi, Dahua Lin, and Yu Qiao. Delving Into the Devils of Bird's-Eye-View Perception: A Review, Evaluation and Recipe. *TPAMI*, 2023. 2

[41] Tianyu Li, Li Chen, Huijie Wang, Yang Li, Jiazhi Yang, Xiangwei Geng, Shengyin Jiang, Yuting Wang, Hang Xu, Chunjing Xu, Junchi Yan, Ping Luo, and Hongyang Li. Graph-based Topology Reasoning for Driving Scenes. *arXiv preprint arXiv:2304.05277*, 2023. 5

[42] Xiaofan Li, Yifu Zhang, and Xiaoqing Ye. DrivingDiffusion: Layout-Guided multi-view driving scene video generation with latent diffusion model. *arXiv preprint arXiv:2310.07771*, 2023. 13

[43] Yanwei Li, Yilun Chen, Xiaojuan Qi, Zeming Li, Jian Sun, and Jiaya Jia. Unifying Voxel-based Representation with Transformer for 3D Object Detection. In *NeurIPS*, 2022. 1, 7

[44] Yinhao Li, Han Bao, Zheng Ge, Jinrong Yang, Jianjian Sun, and Zeming Li. BEVStereo: Enhancing Depth Estimation in Multi-View 3D Object Detection with Temporal Stereo. In *AAAI*, 2023. 7

[45] Yinhao Li, Zheng Ge, Guanyi Yu, Jinrong Yang, Zengran Wang, Yukang Shi, Jianjian Sun, and Zeming Li. BEVDepth: Acquisition of Reliable Depth for Multi-view 3D Object Detection. In *AAAI*, 2023. 5, 15

[46] Yanwei Li, Zhiding Yu, Jonah Philion, Anima Anandkumar, Sanja Fidler, Jiaya Jia, and Jose Alvarez. End-to-end 3D Tracking with Decoupled Queries. In *ICCV*, 2023. 7

[47] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. BEVFormer: Learning Bird's-Eye-View Representation from Multi-Camera Images via Spatiotemporal Transformers. In *ECCV*, 2022. 1, 2, 3, 5, 6, 7, 8, 15

[48] Zhiqi Li, Zhiding Yu, Wenhai Wang, Anima Anandkumar, Tong Lu, and José Manuel Álvarez. FB-BEV: BEV Representation from Forward-Backward View Transformations. *ICCV*, 2023. 2, 4

[49] Ming Liang, Bin Yang, Wenyuan Zeng, Yun Chen, Rui Hu, Sergio Casas, and Raquel Urtasun. PnPNet: End-to-End Perception and Prediction With Tracking in the Loop. In *CVPR*, 2020. 5, 7

[50] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and

C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *ECCV*, 2014. 6, 7

[51] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature Pyramid Networks for Object Detection. In *CVPR*, 2017. 5

[52] Haisong Liu, Yao Teng, Tao Lu, Haiguang Wang, and Limin Wang. SparseBEV: High-Performance Sparse 3D Object Detection from Multi-Camera Videos. *arXiv preprint arXiv:2308.09244*, 2023. 1

[53] Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. PETR: Position Embedding Transformation for Multi-View 3D Object Detection. In *ECCV*, 2022. 1, 3

[54] Yingfei Liu, Junjie Yan, Fan Jia, Shuailin Li, Qi Gao, Tiancai Wang, Xiangyu Zhang, and Jian Sun. PETRv2: A Unified Framework for 3D Perception from Multi-Camera Images. In *ICCV*, 2023. 7

[55] Zhijian Liu, Haotian Tang, Alexander Amini, Xinyu Yang, Huizi Mao, Daniela L Rus, and Song Han. BEVFusion: Multi-Task Multi-Sensor Fusion with Unified Bird's-Eye View Representation. In *ICRA*, 2023. 1

[56] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. *arXiv preprint arXiv:1711.05101*, 2019. 6

[57] B. Mersch, X. Chen, J. Behley, and C. Stachniss. Self-supervised Point Cloud Prediction Using 3D Spatio-temporal Convolutional Networks. In *CoRL*, 2021. 3

[58] Gregory P. Meyer, Ankit Laddha, Eric Kee, Carlos Vallespi-Gonzalez, and Carl K. Wellington. LaserNet: An Efficient Probabilistic 3D Object Detector for Autonomous Driving. In *CVPR*, 2019. 3

[59] Chen Min, Liang Xiao, Dawei Zhao, Yiming Nie, and Bin Dai. Occupancy-MAE: Self-Supervised Pre-Training Large-Scale LiDAR Point Clouds With Masked Occupancy Autoencoders. *TIV*, 2023. 2

[60] Ziqi Pang, Zhichao Li, and Naiyan Wang. SimpleTrack: Understanding and Rethinking 3D Multi-object Tracking. *arXiv preprint arXiv:2111.09621*, 2021. 5

[61] Dennis Park, Rares Ambrus, Vitor Guizilini, Jie Li, and Adrien Gaidon. Is Pseudo-Lidar Needed for Monocular 3D Object Detection? In *ICCV*, 2021. 2, 3

[62] Cody Reading, Ali Harakeh, Julia Chae, and Steven L. Waslander. Categorical Depth DistributionNetwork for Monocular 3D Object Detection. In *CVPR*, 2021. 1

[63] Ravid Shwartz-Ziv and Yann LeCun. To Compress or Not to Compress–Self-Supervised Learning and Information Theory: A Review. *arXiv preprint arXiv:2304.09355*, 2023. 1

[64] Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Ping Luo, Andreas Geiger, and Hongyang Li. DriveLM: Driving with Graph Visual Question Answering. *arXiv preprint arXiv:2312.14150*, 2023. 1

[65] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in Perception for Autonomous Driving: Waymo Open Dataset. In *CVPR*, 2020. 1

[66] Izzeddin Teeti, Salman Khan, Ajmal Shahbaz, Andrew Bradley, and Fabio Cuzzolin. Vision-based Intention and Trajectory Prediction in Autonomous Vehicles: A Survey. In *IJCAI*, 2022. 1

[67] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive Multiview Coding. In *ECCV*, 2020. 2

[68] Wenwen Tong, Chonghao Sima, Tai Wang, Li Chen, Silei Wu, Hanming Deng, Yi Gu, Lewei Lu, Ping Luo, Dahua Lin, and Hongyang Li. Scene as Occupancy. In *ICCV*, 2023. 1, 2, 3, 5, 6, 7

[69] Huijie Wang, Tianyu Li, Yang Li, Li Chen, Chonghao Sima, Zhenbo Liu, Bangjun Wang, Peijin Jia, Yuting Wang, Shengyin Jiang, Feng Wen, Hang Xu, Ping Luo, Junchi Yan, Wei Zhang, and Hongyang Li. OpenLane-V2: A Topology Reasoning Benchmark for Unified 3D HD Mapping. In *NeurIPS Datasets and Benchmarks*, 2023. 5

[70] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. FCOS3D: Fully Convolutional One-Stage Monocular 3D Object Detection. In *ICCV Workshops*, 2021. 1, 2, 3, 6, 7

[71] Tai Wang, Qing Lian, Chenming Zhu, Xinge Zhu, and Wenwei Zhang. MV-FCOS3D++: Multi-View Camera-Only 4D Object Detection with Pretrained Monocular Backbones. *arXiv preprint arXiv:2207.12716*, 2022. 1

[72] Wenhai Wang, Jifeng Dai, Zhe Chen, Zhenhang Huang, Zhiqi Li, Xizhou Zhu, Xiaowei Hu, Tong Lu, Lewei Lu, Hongsheng Li, Xiaogang Wang, and Yu Qiao. InternImage: Exploring Large-Scale Vision Foundation Models with Deformable Convolutions. In *CVPR*, 2023. 6, 7

[73] Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, and Jiwen Lu. DriveDreamer: Towards Real-world-driven World Models for Autonomous Driving. *arXiv preprint arXiv:2309.09777*, 2023. 13

[74] Yue Wang, Vitor Guizilini, Tianyuan Zhang, Yilun Wang, Hang Zhao, , and Justin M. Solomon. DETR3D: 3D Object Detection from Multi-view Images via 3D-to-2D Queries. In *CoRL*, 2021. 7

[75] Chen Wei, Haoqi Fan, Saining Xie, Chao-Yuan Wu, Alan Yuille, and Christoph Feichtenhofer. Masked Feature Prediction for Self-Supervised Visual Pre-Training. In *CVPR*, 2022. 2

[76] Yi Wei, Linqing Zhao, Wenzhao Zheng, Zheng Zhu, Jie Zhou, and Jiwen Lu. SurroundOcc: Multi-Camera 3D Occupancy Prediction for Autonomous Driving. In *ICCV*, 2023. 1

[77] Xinshuo Weng, Jianren Wang, Sergey Levine, Kris Kitani, and Nicholas Rhinehart. Inverting the Pose Forecasting Pipeline with SPF2: Sequential Pointcloud Forecasting for Sequential Pose Forecasting. *arXiv preprint arXiv:2003.08376*, 2020. 3

[78] Xinshuo Weng, Junyu Nan, Kuan-Hui Lee, Rowan McAllister, Adrien Gaidon, Nicholas Rhinehart, and Kris M Kitani. S2Net: Stochastic Sequential Pointcloud Forecasting. In *ECCV*, 2022. 3

[79] Penghao Wu, Xiaosong Jia, Li Chen, Junchi Yan, Hongyang Li, and Yu Qiao. Trajectory-guided Control Prediction for End-to-end Autonomous Driving: A Simple yet Strong Baseline. In *NeurIPS*, 2022. 1

[80] Penghao Wu, Li Chen, Hongyang Li, Xiaosong Jia, Junchi Yan, and Yu Qiao. Policy Pre-training for Autonomous Driving via Self-supervised Geometric Modeling. In *ICLR*, 2023. 2

[81] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised Feature Learning via Non-Parametric Instance Discrimination. In *CVPR*, 2018. 2

[82] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. PointContrast: Unsupervised Pre-training for 3D Point Cloud Understanding. In *ECCV*, 2020. 2

[83] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. SimMIM: A Simple Framework for Masked Image Modeling. In *CVPR*, 2022. 2

[84] Xiangchao Yan, Runjian Chen, Bo Zhang, Jiakang Yuan, Xinyu Cai, Botian Shi, Wenqi Shao, Junchi Yan, Ping Luo, and Yu Qiao. SPOT: Scalable 3D Pre-training via Occupancy Prediction for Autonomous Driving. *arXiv preprint arXiv:2309.10527*, 2023. 2

[85] Chenyu Yang, Yuntao Chen, Hao Tian, Chenxin Tao, Xizhou Zhu, Zhaoxiang Zhang, Gao Huang, Hongyang Li, Yu Qiao, Lewei Lu, Jie Zhou, and Jifeng Dai. BEVFormer v2: Adapting Modern Image Backbones to Bird's-Eye-View Recognition via Perspective Supervision. In *CVPR*, 2023. 1

[86] Honghui Yang, Sha Zhang, Di Huang, Xiaoyang Wu, Haoyi Zhu, Tong He, Shixiang Tang, Hengshuang Zhao, Qibo Qiu, Binbin Lin, Xiaofei He, and Wanli Ouyang. UniPAD: A Universal Pre-training Paradigm for Autonomous Driving. *arXiv preprint arXiv:2310.08370*, 2023. 2, 3

[87] Jia Zeng, Li Chen, Hanming Deng, Lewei Lu, Junchi Yan, Yu Qiao, and Hongyang Li. Distilling Focal Knowledge From Imperfect Expert for 3D Object Detection. In *CVPR*, 2023. 2, 4

[88] Lunjun Zhang, Yuwen Xiong, Ze Yang, Sergio Casas, Rui Hu, and Raquel Urtasun. Learning unsupervised world models for autonomous driving via discrete diffusion. *arXiv preprint arXiv:2311.01017*, 2023. 13

[89] Tianyuan Zhang, Xuanyao Chen, Yue Wang, Yilun Wang, and Hang Zhao. MUTR3D: A Multi-camera Tracking Framework via 3D-to-2D Queries. In *CVPR*, 2022. 5, 7

[90] Yunpeng Zhang, Zheng Zhu, Wenzhao Zheng, Junjie Huang, Guan Huang, Jie Zhou, and Jiwen Lu. BEVerse: Unified Perception and Prediction in Birds-Eye-View for Vision-Centric Autonomous Driving. *arXiv preprint arXiv:2205.09743*, 2022. 5, 7

[91] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In *ICLR*, 2020. 5

Appendix

A. Discussions

Towards a better understanding of our work, we supplement intuitive questions that one might raise.

Q1: *What is the relationship between ViDAR and world models?*

In general, ViDAR could be deemed as a world model - predicting the future world conditioned on observations and actions. However, it distinguishes itself from existing world models for autonomous driving [24, 42, 73, 88]. Unlike these models, which operate within the same modality for both inputs and outputs (*e.g.*, image-in & image-out or LiDAR-in & LiDAR-out), ViDAR for the first time bridges different modalities. It leverages historical visual sequences as inputs to predict future point clouds. By utilizing ViDAR, it becomes possible to generate various future point clouds, conditioned on different future ego-vehicle motions, using visual inputs. This holds significant potential in training vision autonomy in 3D.

Q2: *Why use point clouds (LiDAR) as outputs, instead of future images?*

Compared to images, point clouds offer a highly precise depiction of the 3D environment, effectively capturing scene structure, object positions, and geometric properties. This detailed representation proves advantages in various 3D tasks, including perception, reconstruction, and rendering. Consequently, in ViDAR, we opt to employ point clouds as the prediction target. This choice enables the model to extract 3D geometry from visual inputs, thereby empowering downstream models.

Q3: *What are potential applications and future directions of ViDAR?*

In our work, we have demonstrated the effectiveness of ViDAR as a pre-training approach for enhancing downstream end-to-end autonomous driving models [7, 28]. Additionally, considering its capability as a world model, there is significant promise in utilizing ViDAR as a simulator for model-based reinforcement learning, thereby bolstering the decision-making abilities of vision-based autonomous agents. The application of ViDAR in this context opens up avenues for future research. For instance, to facilitate its scalability, investigations into the utilization of Image-LiDAR sequences from diverse datasets are necessary. Furthermore, exploring the combination of ViDAR with other single-modality world models to create more favorable data simulations presents intriguing prospects for future inquiry.

B. Implementation Details

Latent Rendering. We now delve into the specific implementation details of the Latent Rendering operator. As shown in Figure 7, the Latent Rendering operator comprises a total of 4 steps. In the first step, given the BEV embeddings from the Latent Rendering, we use a projection layer with output channels of G to estimate independent probability maps:

$$\mathbf{p} = \texttt{Projection}(\mathcal{F}_{\text{bev}}), \tag{8}$$

where G is the group number for multi-group Latent Rendering. To simplify, we focus on where $G = 1$. In cases of multiple groups, we can easily divide $\mathcal{F}_{\text{bev}}$ into G parts along the channels, and apply Latent Rendering on each part accordingly.

In the second step, we compute the conditional probability of each BEV grid along its respective ray. As shown in step-(a.) of Figure 7, for the i-th BEV grid with coordinates $\mathbf{g}^{(i)} = \{x_i, y_i\}$, we begin by casting a ray from the origin point $\mathbf{o}$, typically the center of BEV feature maps, towards the target BEV grid. The direction of the ray is determined as:

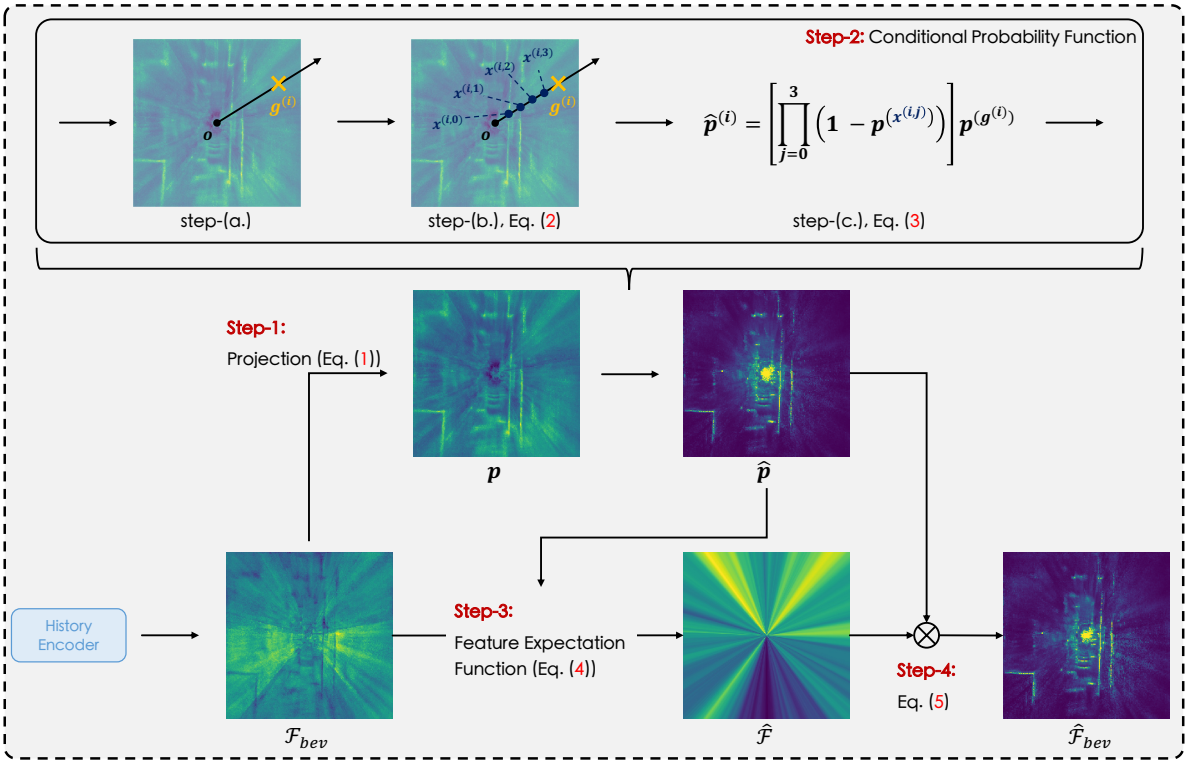

Figure 7. **Detailed architecture of the Latent Rendering operator.** We only show the case of single-group for simplicity. Given the visual BEV embedding $\mathcal{F}_{\text{bev}}$ from the History Encoder, the Latent Rendering operator performs a series of steps. First, it employs a projection layer to estimate the independent probability map, $\mathbf{p}$. Subsequently, the conditional probability function generates the conditional probability of each BEV grid, denoted as $\hat{\mathbf{p}}^{(i)}$. Then, we compute ray-wise features using a feature expectation function, and finally, the resulting ray-wise features are multiplied with the conditional probability map to yield the geometric feature space.

$\mathbf{d}^{(i)} = (\mathbf{g}^{(i)} - \mathbf{o})/(||\mathbf{g}^{(i)} - \mathbf{o}||_2)$. Subsequently, we collect a set of prior waypoints along the ray, which are closer to the origin point in distance compared to BEV grid i. These waypoints are selected at uniform distance intervals, termed as λ, as:

$$\mathbf{x}^{(i,j)} = \mathbf{o} + j\lambda\mathbf{d}^{(i)}, \text{ where } j\lambda < ||\mathbf{g}^{(i)} - \mathbf{o}||_2. \tag{9}$$

Here, j represents the index of different waypoints, and the condition $j\lambda < ||\mathbf{g}^{(i)} - \mathbf{o}||_2$ ensures that the distance of these waypoints from the origin point is smaller than that of the corresponding BEV grid. An example is depicted in step-(b.) of Figure 7, where we collect 4 prior waypoints with coordinates as $\mathbf{x}^{(i,j)}$ where $j \in \{0, 1, 2, 3\}$ for the i-th BEV grid. Moving on to step-(c.) of Figure 7, we compute the conditional probability of the target BEV grid based on the independent probability of its prior waypoints. In general, the conditional probability function is defined as:

$$\hat{\mathbf{p}}^{(i)} = \left[\prod_j (1 - \mathbf{p}^{(\mathbf{x}^{(i,j)})})\right] \mathbf{p}^{(\mathbf{g}^{(i)})}, \tag{10}$$

where $\mathbf{p}^{(:)}$ is the bilinear interpolation function to compute the corresponding probability of the position $(:)$ from $\mathbf{p}$.

In the third step, we compute the ray-wise features according to the BEV embedding and conditional probability map:

$$\hat{\mathcal{F}}^{(i)} = \sum_k \hat{\mathbf{p}}^{(i,k)} \mathcal{F}_{\text{bev}}^{(k)}, \text{ where } \mathbf{d}^{(k)} = \mathbf{d}^{(i)}, \tag{11}$$

where k indexes those BEV grids in the same direction as the i-th BEV grid.

As illustrated in Figure 7, after the feature expectation function, all BEV grids lying on the same ray share the same global feature embeddings. Therefore, finally, in order to highlight those BEV grids with higher conditional probability, we multiply the ray-wise features and the conditional probability map to obtain the geometric feature responses:

$$\hat{\mathcal{F}}_{\text{bev}} = \hat{\mathbf{p}} \cdot \hat{\mathcal{F}}. \tag{12}$$

In our concrete implementation, we set G and λ as 16 and 1, respectively.

C. Additional Ablative Studies

C.1. Structure of Latent Rendering

In Table 12, we investigate the effectiveness of conditional probability function (Eq. (10), Step-2 in Figure 7) and feature expectation function (Eq. (11), Step-3 in Figure 7). For context, we use BEVFormer-small [47] with ImageNet-cls pre-training as the baseline. We evaluate different functions by first pre-training the baseline using ViDAR under various structures and then comparing the downstream 3D detection performance. The baseline result, 40.20% NDS, is obtained by pre-training the model using the straightforward pipeline, mentioned in Section 3.2 of the main paper, which comprises the History Encoder, Future Decoder, and differentiable ray-casting.

Cond. Prob. Func., Eq. (10)	Feat. Exp. Func., Eq. (11)	NDS (%)
-	-	40.20
√	-	47.34
√	√	**47.58**

Table 12. **Ablation on each component of Latent Rendering design.** "Cond. Prob. Func." and "Feat. Exp. Func." represent the conditional probability function and the feature expectation function, respectively.

To evaluate the effectiveness of the conditional probability function solely, we directly multiply the BEV embeddings $\mathcal{F}_{\text{bev}}$ with the aggregated conditional probability map $\hat{\mathbf{p}}$ to obtain the features for future point cloud prediction. As illustrated in the second row of Table 12, the conditional probability function greatly alleviates the ray-shaped feature issue, and brings a 7.14% NDS improvement compared to the simple baseline. Then, by introducing the feature expectation function, the Latent Rendering operator integrates the entire ray-wise feature, which further brings downstream performance improvements.

C.2. Future Predictions

In this study, we aim to showcase the benefits of future prediction in downstream tasks that involve temporal modeling. We conduct ablation studies on the task of Multi-Object Tracking, as it necessitates the model to associate 3D objects across different frames, thereby reflecting its temporal modeling capabilities. For our experiments, we use the first stage of UniAD [28] as the tracking model, and assess its performance on downstream tracking after being pre-trained with ViDAR using varying frames of future point cloud supervision.

No. of Future Supervision	Detection NDS (%) ↑	Tracking AMOTA (%) ↑
0	54.04	42.12
1	53.73	43.79
3	53.60	44.74
6	53.81	**45.10**

Table 13. **Ablation on future supervision.** More future supervisions during the pre-training stage bring consistent improvements in tracking performance.

As shown in Table 13, we evaluate four settings which utilize "0", "1", "3", and "6" future frames to supervise ViDAR when visual point cloud forecasting pre-training. The setting labeled as "0 future supervision" implies that we solely utilize ViDAR to reconstruct LiDAR point clouds for the current frame, without any future prediction and supervision. As illustrated, we observe consistent improvements in tracking performance when using more future frames for supervision. This observation highlights the beneficial impact of visual point cloud forecasting on the temporal modeling capabilities of downstream models.

C.3. Pre-train Structure

In Table 14, we investigate the effect of ViDAR pre-training on different components of downstream BEV encoders. We achieve this by loading different sets of pre-trained model parameters during the downstream fine-tuning process. For context, in this ablation study, we use BEVFormer-small as the downstream model for 3D detection. The baseline performance is 44.11% NDS.

2D Backbone	FPN-neck	View-Transform	NDS (%)
-	-	-	44.11
√	√	-	44.74
√	√	√	47.58

Table 14. **Ablation on pre-training components.** ViDAR mostly benefits the view-transformation part of visual BEV encoders.

As illustrated, the improvements primarily stem from the pre-training of the view-transformation component, which is responsible for extracting BEV features from perspective multi-view image features. It is reasonable considering that the view-transformation module plays a crucial role in correlating 2D features with 3D geometries and scene structures, as discussed in previous works [45]. Furthermore, this highlights the compatibility of ViDAR pre-training with any advanced

2D image pre-training techniques, which could lead to consistent performance improvements when combined with more advanced image backbones, as demonstrated in Table 3 and Table 4 of the main paper.

D. Qualitative Results

D.1. Latent Rendering

Figure 8 presents the effectiveness of the Latent Rendering operator in formulating geometric features from visual sequence inputs. Each pair of images in Figure 8 displays the ground-truth point cloud on the left and visualizes the features, denoted as $\hat{\mathcal{F}}_{\text{bev}}$, on the right. As depicted, ViDAR successfully captures the 3D geometry from multi-view visual sequences and effectively extracts geometric features that accurately represent the underlying 3D world in the latent space.

Next, in Figure 9, we compare the BEV features, $\mathcal{F}_{\text{bev}}$, extracted by BEV encoders pre-trained using the differentiable ray-casting baseline (depicted in the middle) and our ViDAR (depicted on the right). As illustrated, the ray-casting baseline encounters the issue of ray-shaped features after pre-training. In contrast, our ViDAR, thanks to the inclusion of Latent Rendering during the pre-training procedure, effectively highlights the responses of BEV grids preserving geometric information. Consequently, it extracts discriminative features after pre-training, which, in turn, benefits downstream fine-tuning.

D.2. Visual Point Cloud Forecasting

In Figure 10, we provide visual examples of ViDAR for predicting future point clouds based on historical visual images. The historical visual inputs captured within a 1-second timeframe are displayed in the upper portion, while the corresponding predicted future point clouds spanning 3 seconds are shown in the lower portion.

In the first row, we present an example of the ego-vehicle executing a left turn. As observed, ViDAR adeptly captures the related position and orientation between the ego-vehicle and the parked blue bus in its future predictions. Moving on to the second and third rows, we illustrate instances where ViDAR successfully captures the relative motions between the ego-vehicle and other moving objects, such as the yellow bus in the second row and the white car in the third row. By analyzing the LiDAR outputs, ViDAR accurately understands that moving objects exhibit faster movement compared to the ego-vehicle. Consequently, it estimates their positions with increasing relative distances as time progresses. Then, the fourth row showcases an example of the ego-vehicle executing a right turn. This case demonstrates ViDAR's effective modeling of the road map based on the historical visual sequences. It is important to note that all LiDAR point cloud visualizations are presented within the coordinate space of the ego-vehicle, with the ego-vehicle situated at the center of the 3D space.

Additionally, in Figure 11, we demonstrate the capability of ViDAR to simulate various future point clouds based on specific ego-vehicle motions, such as turning left, going straight, and turning right. This showcases the potential of ViDAR as a visual world model for autonomous driving, where it utilizes visual image inputs to generate simulated future point clouds. Such simulations can be valuable for model-based reinforcement learning for training vision autonomy in an unsupervised manner.

D.3. End-to-end Autonomous Driving

Lastly, in Figure 12, we present a comparison between UniAD with and without ViDAR pre-training for end-to-end autonomous driving. As illustrated, the inclusion of ViDAR pre-training enables UniAD to generate more precise future trajectories for other moving objects (highlighted in **red circles**). This improved prediction accuracy plays a crucial role in enhancing the planning process for safety-critical end-to-end autonomous driving scenarios.

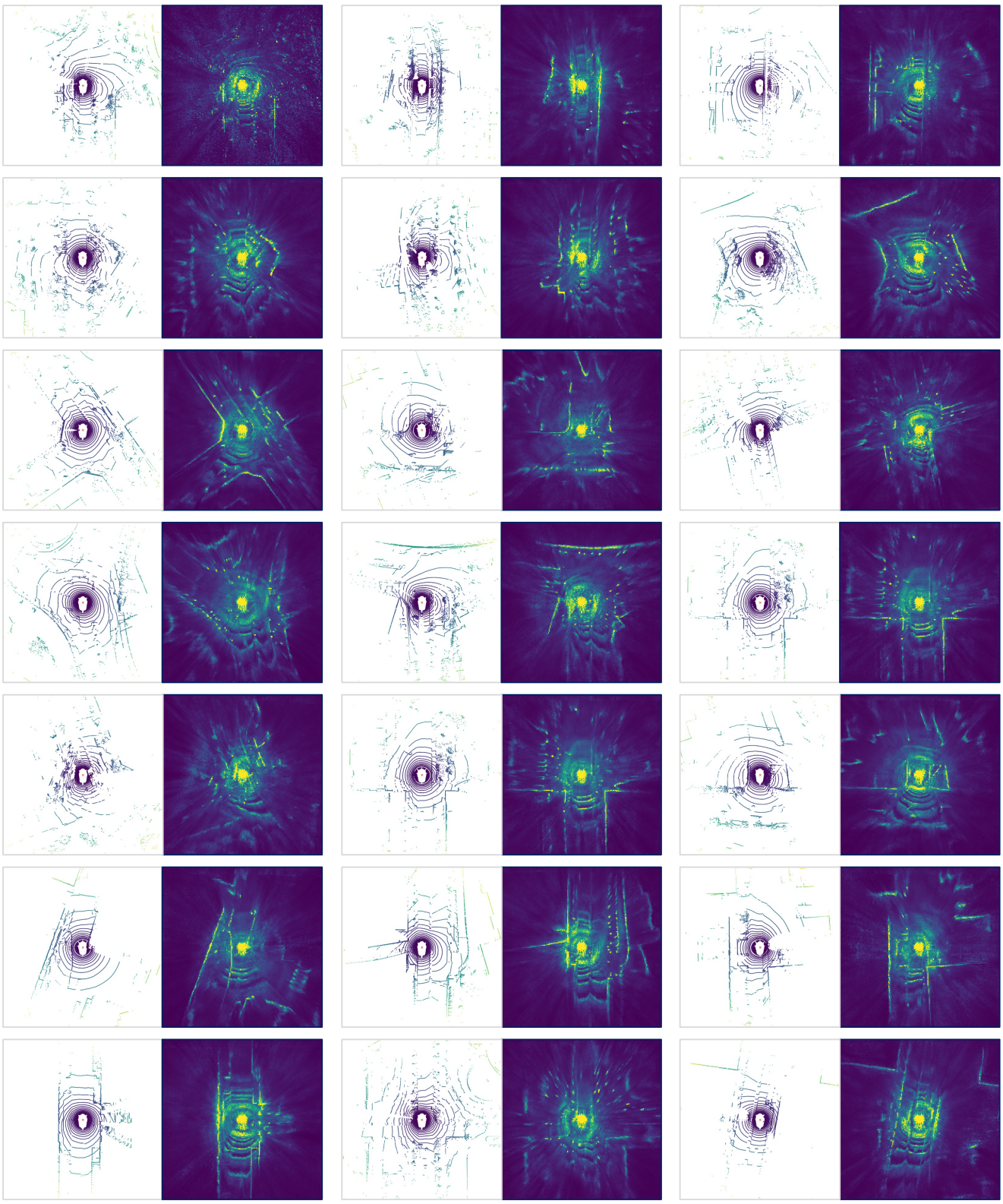

Figure 8. **Visualization of geometric features from the Latent Rendering operator.** In each pair, we present the ground-truth LiDAR point clouds on the left and the BEV features produced by ViDAR from multi-view image inputs on the right. It is evident that, utilizing the Latent Rendering operator, ViDAR successfully captures the underlying 3D geometry in the latent space, resulting in precise descriptions of the 3D world through feature responses.

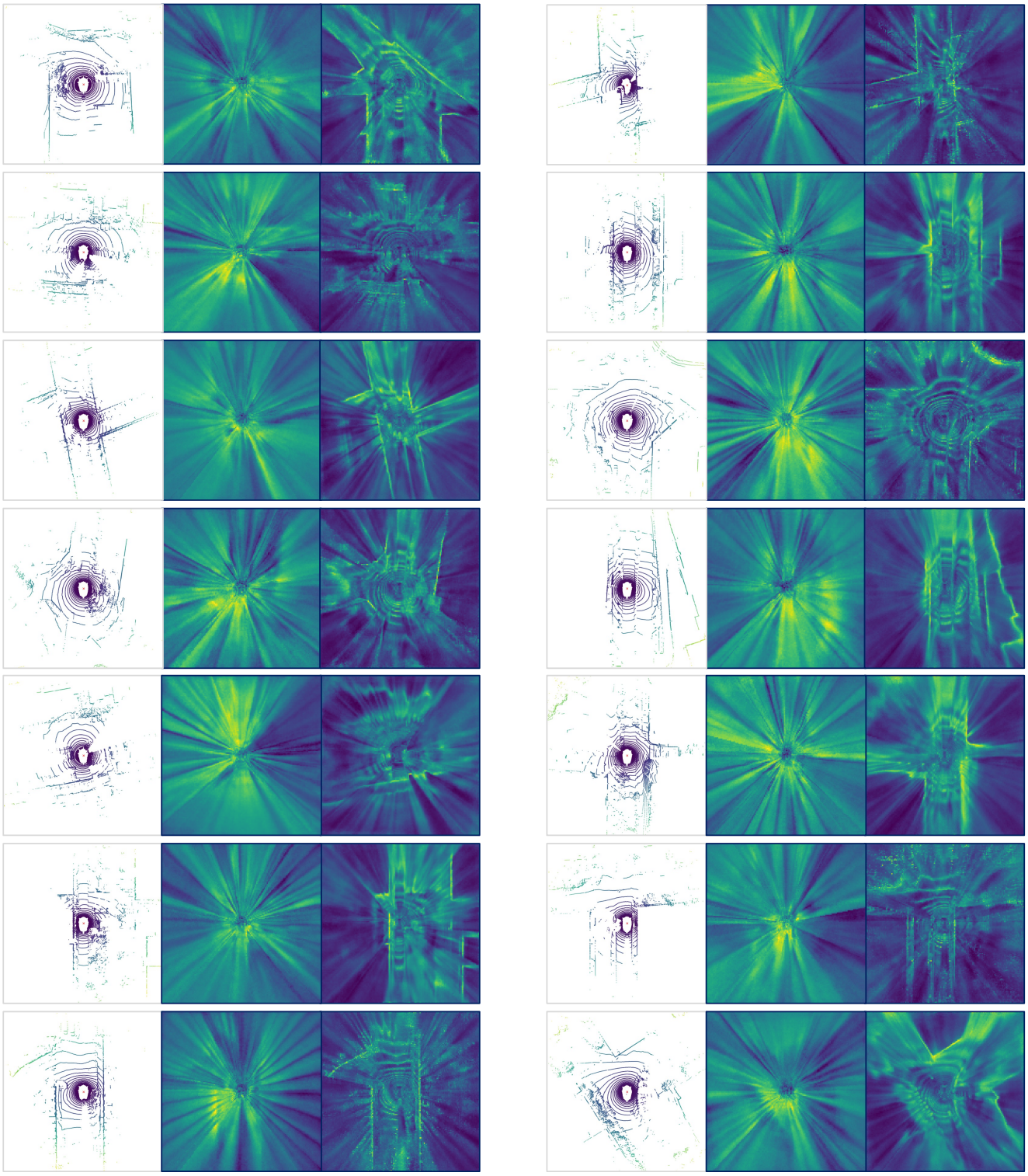

Figure 9. **Visualization of BEV features from visual BEV encoder pre-trained by the differentiable ray-casting baseline and ViDAR with Latent Rendering operator.** In each triplet, we present the ground-truth point cloud on the left, the features pre-trained by the differentiable ray-casting baseline in the middle, and the features pre-trained by ViDAR on the right. As illustrated, the integration of the Latent Rendering operator in ViDAR proves advantageous as it successfully mitigates the occurrence of ray-shaped features during visual point cloud forecasting pre-training. Consequently, ViDAR empowers the BEV encoders to extract informative and discriminative BEV features from visual sequence inputs.

Figure 10. **Qualitative results of ViDAR for visual point cloud forecasting on nuScenes validation set.** Top: historical visual inputs in 1 second; bottom: future point cloud predictions in 3 seconds. The first, second, and third rows provide examples of modeling relative motions between the ego-vehicle and other parked or moving objects (highlighted in **red circles**); the fourth row shows the future estimation where the ego-vehicle is turning right. As illustrated, ViDAR effectively captures the information of 3D geometry and temporal dynamics, and thus correctly estimates future point clouds from visual sequence inputs. All point cloud visualizations are under the ego coordinates.

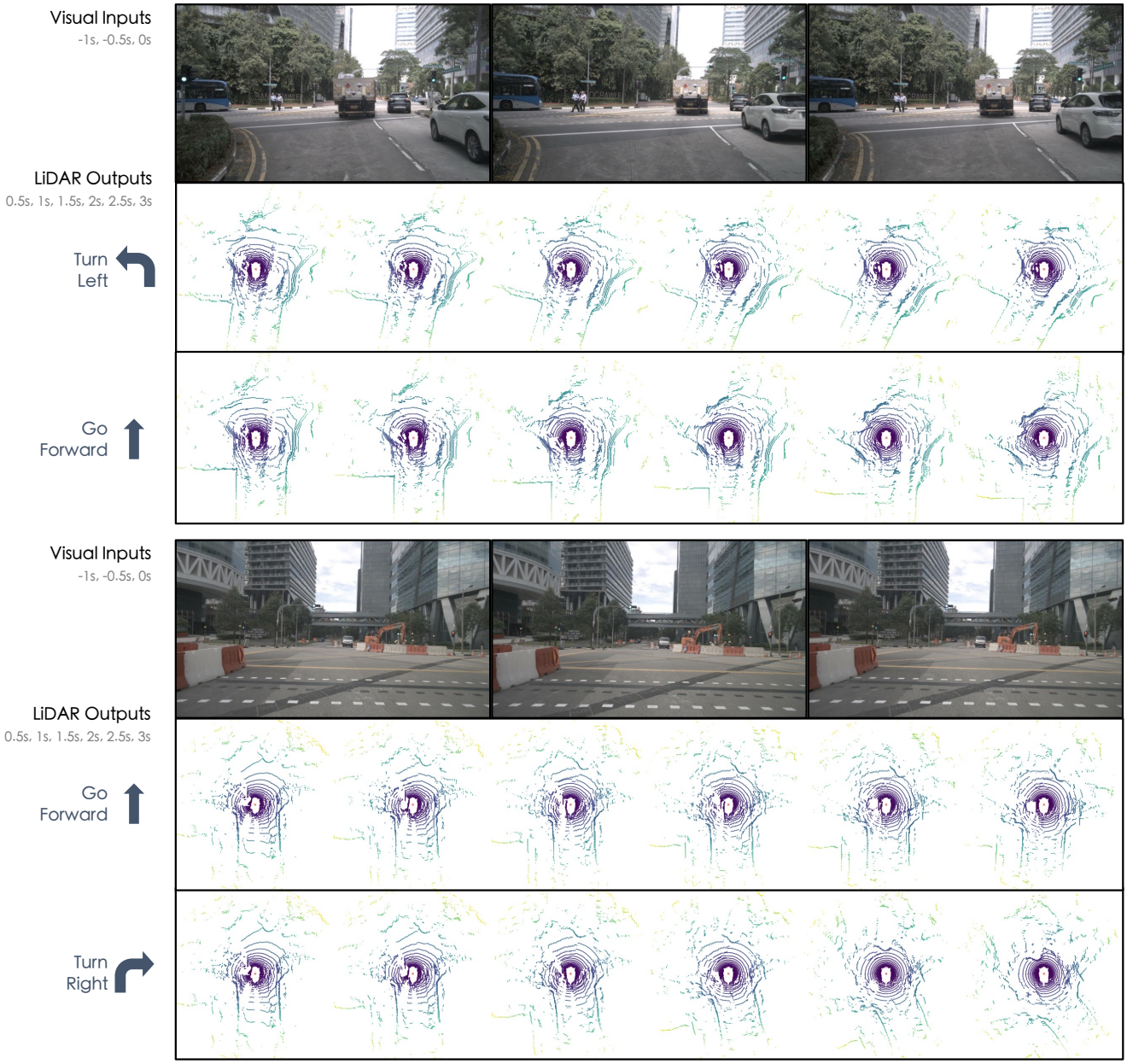

Figure 11. **Qualitative results of ViDAR conditioned on different future ego-vehicle motion conditions.** ViDAR can simulate different future point clouds based on the specific future ego-vehicle conditions. All point cloud visualizations are under the ego coordinates.

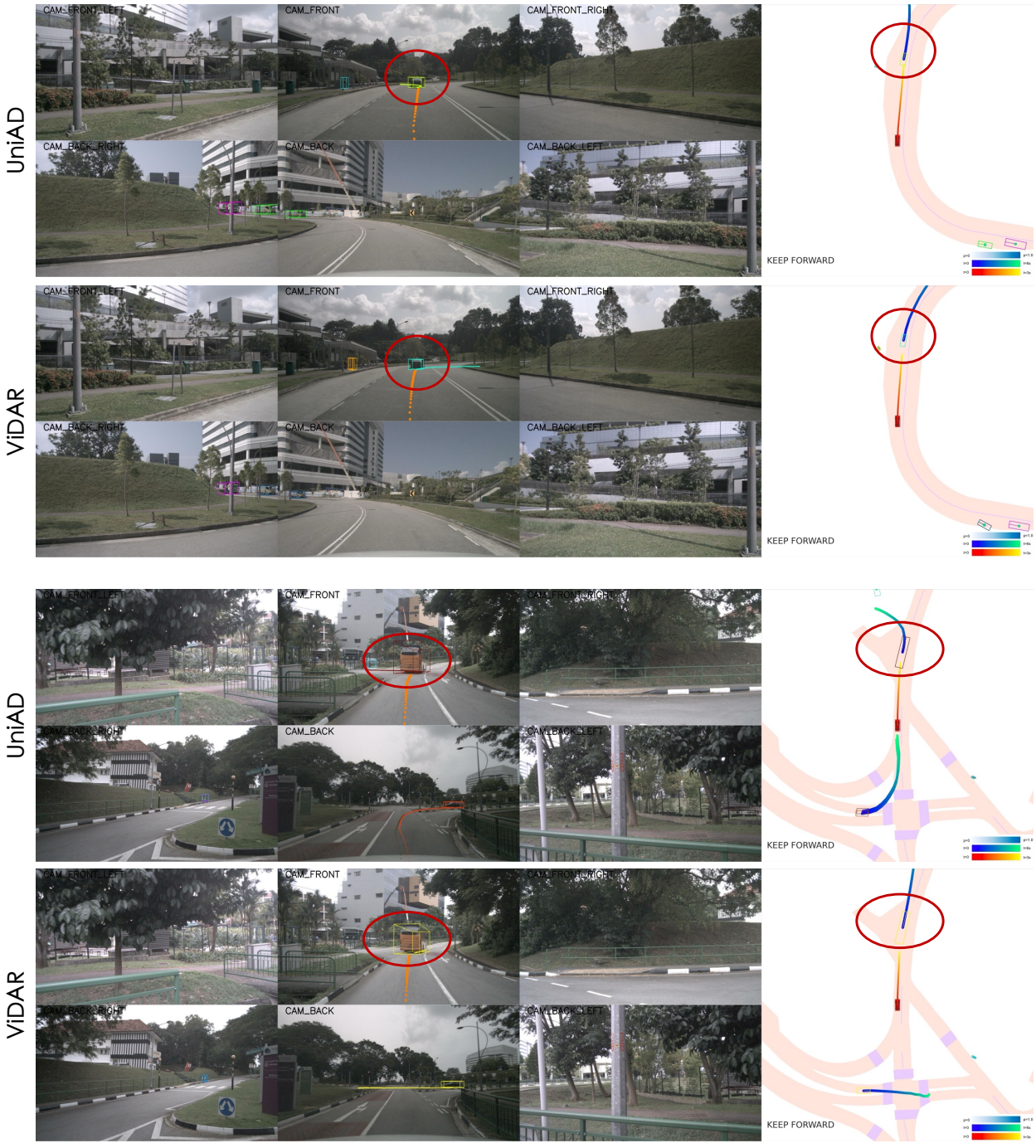

Figure 12. **Qualitative comparisons between UniAD with and without ViDAR pre-training.** ViDAR effectively models the temporal dynamics by estimating future points during pre-training, which in turn improves UniAD in predicting the future motion of moving objects.