LiFi: LiDAR Generation from Multi-View Images via Geometric and Semantic Collaborative Guidance

Sizhuo Zhou[1,2], **Xiaosong Jia**[4†], **Fanrui Zhang**[1,2], **Qifeng Li**[3], **Junjie Li**[1,2], **Zirui Wang**[1,2], **Yukang Feng**[2], **Yu Hong**[2,3], **Shaofeng Zhang**[3], **Wenlong Liao**[3], **Tao He**, **Juyong Zhang**[1], **Junchi Yan**[2,3†]

1. University of Science and Technology of China
2. Shanghai Innovation Institute
3. Shanghai Jiao Tong University
4. Fudan University

Abstract

Generating LiDAR point clouds from multi-view camera images is a valuable task with applications in controllable data synthesis, cross-modal simulation and various perception tasks. However, most existing methods are not specifically designed for this task, and the quality of their generated LiDAR remains limited. Moreover, these methods typically focus only on semantic cues in images or generate LiDAR from a monocular camera image. In this work, we introduce **LiFi** (**Li**DAR **F**rom **i**mages), a dedicated framework for generating visually and geometrically aligned LiDAR scenes from multi-view camera images. LiFi guides the latent diffusion process through two complementary branches: geometry and semantics. The geometry branch constructs scene features from images through a reverse sampling algorithm and an uncertainty-aware geometric encoder combined with DepthAnything3, whereas the semantic branch provides high-level semantic representations via a cross-view interaction module. We further introduce a dual-branch balanced classifier-free guidance (CFG) strategy, which enhances the model's conditional generation capability while preserving the independent learning and collaborative controllability of the two branches. Extensive experiments demonstrate that LiFi outperforms state-of-the-art methods in generating high-fidelity LiDAR scenes. The code will be publicly available at: https://github.com/szzhou88/LiFi.

1 Introduction

The rapid progress of autonomous driving (AD) has drawn widespread attention [18–20]. LiDAR sensors, which provide precise 3D structural information, have become a cornerstone of modern AD systems [30, 32]. Nevertheless, acquiring large-scale, high-quality LiDAR data could be expensive and the long-tail data captured by consumer devices might not employ LiDAR due to cost control of the product, limiting the robustness and scalability of data-driven methods. To mitigate this, pioneering LiDAR generation approaches [14, 52, 57] have been introduced, but they mainly focus on unconditional generation and suffer from limited realism because they fail to exploit the diverse contextual cues present in real-world scenes. Recently, conditional LiDAR point cloud generation [24, 28, 49, 51] has emerged, leveraging inputs such as text, maps, and semantic layouts to achieve notable gains in both the photorealism and controllability of synthesized LiDAR scenes.

In this work, we study the conditional generation of LiDAR scenes from multi-view camera images. Cameras, owing to their low cost and ease of deployment, have become a standard sensor in mod-

† Correspondence Authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

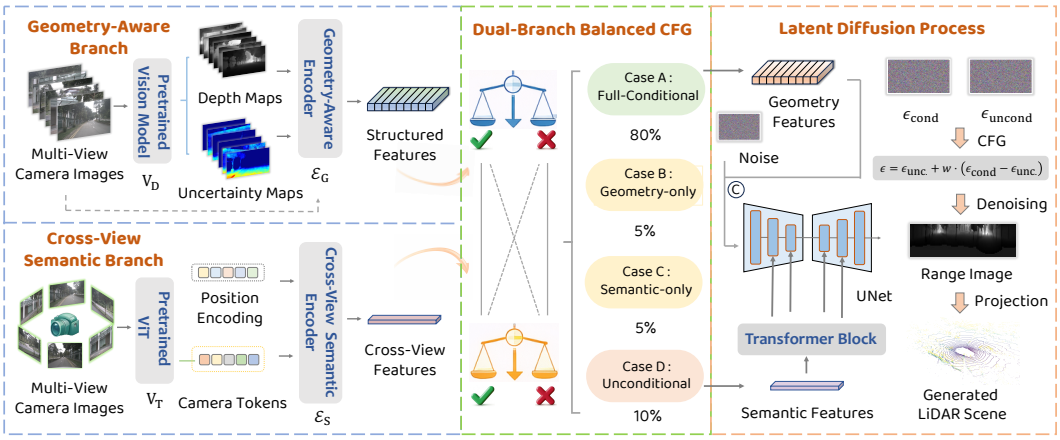


Figure 1: **Overview of the LiFi framework.** The framework takes real multi-view camera images as input and is capable of generating 3D LiDAR scenes with high fidelity that are consistent with the visual information. LiFi mainly comprises the following key designs: (i) geometry–semantic decoupled conditioning branches; (ii) a dual-branch balanced classifier-free guidance module; and (iii) a latent diffusion network integrated with Transformer blocks.

ern autonomous systems. Generating LiDAR point clouds from such onboard multi-view images therefore holds broad application potential, where it can complement high-quality 3D perception under cost-sensitive sensor setups, and open up new possibilities for cross-modal simulation [11], data augmentation [5], and sensor completion [6, 56]. However, this formulation remains relatively under-explored. A major challenge is the cross-modal gap between images and LiDAR. RGB images provide dense 2D appearance information, while LiDAR captures sparse yet precise 3D geometric measurements, making it difficult to establish accurate correspondences between the two heterogeneous representations. Furthermore, although multi-view cameras provide richer observations, different viewpoints also introduce variations in field of view, occlusion patterns, and semantic redundancy. As a result, existing methods [40] have only preliminarily explored images as conditioning inputs, focusing on semantic cues without explicitly enforcing geometric alignment between modalities, and are often limited to monocular inputs that lack comprehensive spatial awareness.

To address these challenges, we propose **LiFi**, a dedicated framework for generating high-fidelity LiDAR scenes from multi-view camera images. LiFi guides the latent diffusion process of LiDAR scenes, represented as range images, through two decoupled yet complementary branches: geometry and semantics. In the geometry branch, we first leverage DepthAnything3 [31] to estimate per-pixel depth and uncertainty maps from the input images. A range-view-centered reverse sampling algorithm then is designed to align these dense predictions, together with the original RGB images, to the LiDAR scanning pattern, producing depth, confidence, and appearance conditions. These conditions are subsequently encoded into structured geometric features via an uncertainty-modulated encoder that dynamically adjusts feature expressiveness based on local depth confidence. In the semantic branch, we extract high-level visual tokens and fuse them with patch, view, and camera extrinsic encodings. A cross-view attention module then aggregates global semantic context to capture long-range dependencies and cross-view consistency. Furthermore, we introduce a dual-branch balanced classifier-free guidance (CFG) strategy. By jointly training with full conditional, single-branch-conditional, and unconditional settings, this strategy not only enhances the model's conditional guidance capability, but also improves the stability and effectiveness of collaborative learning between the two branches.

We primarily compare LiFi with the state-of-the-art LiDAR generation models on the nuScenes dataset. We also compare it with existing frameworks capable of performing camera-to-LiDAR generation. To further evaluate the applicability of LiFi, we also compare it with several representative baseline methods on the KITTI-360 dataset [7]. Experimental results demonstrate the superior performance of our method in generating high-fidelity LiDAR scenes with geometric consistency and semantic alignment. Furthermore, we apply LiFi to a downstream object detection task. Although this does not directly demonstrate that LiFi-generated scenes can independently support perception performance

comparable to that of real LiDAR data, it nevertheless provides evidence of the effectiveness of the proposed generation framework.

In summary, our contributions are as follows:

- We propose LiFi, which, to the best of our knowledge, is the first dedicated framework capable of generating high-fidelity LiDAR scenes with geometric–visual alignment from multi-view images;
- We carefully design a range-view-centered reverse sampling algorithm, an uncertainty-aware geometric encoder, and a cross-view semantic encoder to achieve precise cross-modal alignment at the geometric level and effective global semantic guidance;
- We introduce a dual-branch balanced classifier-free guidance strategy, which effectively enhances the model's conditional guidance capability and collaborative learning ability;
- We conduct extensive evaluations of LiFi on the nuScenes dataset for both high-fidelity LiDAR generation and downstream detection tasks, demonstrating the effectiveness of our method.

2 Related Work

Generative Models. Generative models synthesize realistic and diverse new samples by learning the underlying data distribution. Early generative frameworks include GANs [8] and VAEs [23]. GANs can generate realistic images but suffer from issues such as training instability and mode collapse. VAEs provide an explicit probabilistic modeling formulation, but their generated results tend to be overly smooth. Autoregressive models [48] achieve accurate generation by progressively modeling conditional distributions, yet their sequential sampling process incurs high computational overhead. Diffusion models (DMs) [12, 22, 36, 42, 43, 45–47] have demonstrated outstanding performance in image synthesis. Subsequent research has introduced latent diffusion models (LDMs) [41], which perform the diffusion process in the latent space, thereby significantly improving generation efficiency. Meanwhile, they have also facilitated the advancement of conditional generation [35, 39, 53].

LiDAR Scene Generation. LiDAR scene generation aims to synthesize 3D point cloud representations with realistic geometric structures and scanning patterns. Caccia et al. [1] were the first to adopt the range image representation and perform generation using VAEs and GANs. Subsequently, LiDARGen [57] proposed a score-based model for LiDAR generation. UltraLiDAR [52] voxelizes LiDAR point clouds and combines this representation with VQ-VAE for generation. R2DM [33], built upon DDPMs, improves the fidelity of generated point clouds. RangeLDM [14] enables fast LiDAR data generation by performing diffusion in a compact latent space. DriveLiDAR4D [3] improves LiDAR generation quality by conditioning on three modalities: road sketches, scene descriptions, and object priors. UniScene [24] introduces occupancy as an intermediate representation, thereby reducing the difficulty of generating complex LiDAR scenes. OpenDWM [34] is built upon the DiT architecture for video generation and further extends it to LiDAR scenes. LiDARCrafter [28] parses natural language inputs into scene graphs to guide generation. LaGen [55] proposes an autoregressive framework for generating long-horizon LiDAR scenes.

Camera-to-LiDAR Generation. Camera-to-LiDAR generation refers to generating 3D LiDAR scenes from camera images. Unlike other conditional LiDAR generation models, research on directly synthesizing LiDAR data from camera images remains relatively limited. LiDM [40] proposes a multimodal conditional framework for LiDAR scene generation and also provides a preliminary exploration of the camera-to-LiDAR task by extracting semantic information from camera images. However, it does not investigate how to align the image and LiDAR modalities, which differ substantially from each other, and thus its generation quality remains limited. X-Drive [51] proposes a framework for the joint generation of LiDAR point clouds and multi-view camera images, taking both LiDAR point clouds and camera images as inputs and generating the two modalities simultaneously. It can be adapted to the camera-to-LiDAR task. However, its core framework is not specifically designed for the image-only input setting, and therefore does not exhibit clear performance advantages in this scenario. Moreover, it incorporates additional text prompts and 3D bounding box information to assist the generation process. Other frameworks for the joint generation of LiDAR point clouds and camera images include HoloDrive [50] and Genesis [10], but they do not provide a camera-to-LiDAR mode. Our proposed method addresses the above challenges.

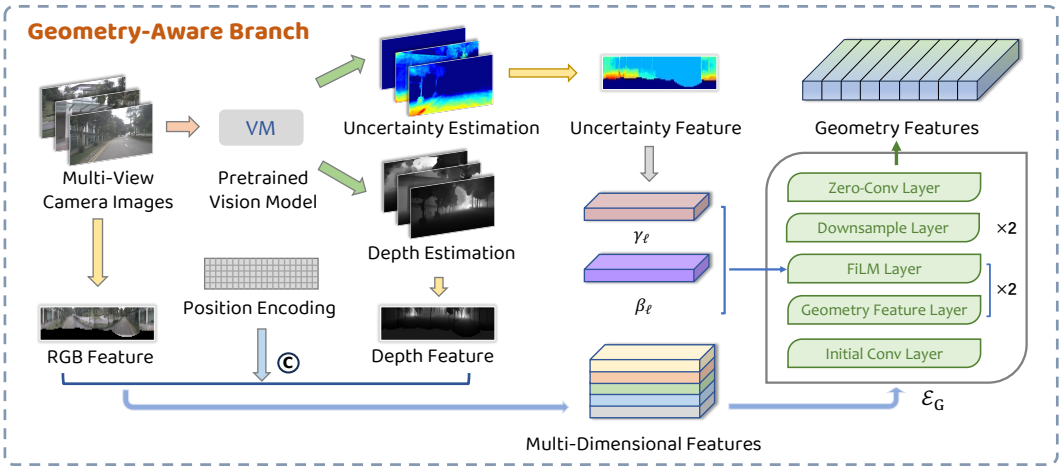


Figure 2: **Geometry-Aware Branch** extracts structured geometric features from raw images via a range-view-centered reverse sampling algorithm and an uncertainty-modulated geometric encoder, thereby guiding the latent diffusion process.

3 Method

3.1 Preliminaries

Range Image Representation. We adopt the range image as the basic representation of LiDAR data. Unlike directly processing sparse and unordered 3D point sets, it exploits the sensor's scanning pattern to encode point clouds into a regular 2D array. For a point $p = (x, y, z) \in \mathbb{R}^3$, we first transform it from the Cartesian coordinate system to a corrected spherical coordinate system [14]:

$$r = \sqrt{x^2 + y^2 + (z - h_j)^2}, \quad \theta = \arctan(y, x), \quad \phi = \phi_j, \tag{1}$$

where r denotes the depth value, θ is the azimuth angle, and ϕ is the elevation angle; h_j and ϕ_j denote the sensor height and elevation angle estimated by Hough voting. We then map this spherical representation to the range-view space and apply normalization, thereby obtaining the corresponding range image. This representation transformation is reversible.

Adopting the range image representation not only transforms the data into a 2D representation but also ensures that each pixel corresponds to a specific azimuth and elevation direction of the target LiDAR sensor, thereby preserving the sensor's structured angular sampling topology and inherent sparsity. Furthermore, given a calibrated sensor model, the generated 2D representation can be deterministically converted back into a 3D point cloud, improving generation efficiency.

Latent Diffusion Model for LiDAR Generation. We adopt a latent diffusion model (LDM) [41] for conditional LiDAR data generation. The framework consists of a VAE and a UNet: the former compresses the range image into a latent space, while the latter performs diffusion modeling in the latent space. Specifically, given an input range image $I \in \mathbb{R}^{H_r \times W_r \times 2}$, the encoder first maps it to a latent representation $\hat{z} \in \mathbb{R}^{h_r \times w_r \times c_r}$, and the decoder then reconstructs the range image from this latent variable as $\hat{I} = G(\hat{z})$. During the diffusion process, the model starts from a clean latent variable z_0 and progressively injects Gaussian noise into it to obtain a noisy latent variable z_t. During training, the UNet takes z_t, the timestep t, and the encoded conditional information c as inputs, and predicts the noise at the current step. The training objective is defined as:

$$\mathcal{L}_{LDM} = \mathbb{E}_{\epsilon_t \in \mathcal{N}(0,1), t \in \mathcal{U}[0,T], c} \left[\left\| \epsilon_t - \epsilon_\theta \left(z_t; t, c \right) \right\|^2 \right], \tag{2}$$

where ϵ_θ denotes the learnable noise prediction network. During inference, the model starts from random Gaussian noise and iteratively denoises it to generate the latent range-view representation.

3.2 Overall Framework

Given the multi-view camera images of a driving scene, $V \in \mathbb{R}^{N_v \times H_v \times W_v \times 3}$, along with the corresponding camera extrinsics $M = [R|t] \in \mathbb{R}^{3\times4}$ and camera intrinsics $K \in \mathbb{R}^{3\times3}$, our goal is to generate a 3D LiDAR scene that is geometrically and perceptually aligned with the input. To achieve this goal, we first extract the following three types of visual priors from multi-view camera inputs using a pre-trained vision foundation model: (i) pixel-wise relative depth estimation in the camera image space: $D \in \mathbb{R}^{N_v \times H_v \times W_v}$; (ii) the corresponding uncertainty maps of depth estimation $U \in \mathbb{R}^{N_v \times H_v \times W_v}$; and (iii) camera semantic tokens $S \in \mathbb{R}^{N_v \times l_s \times c_s}$. Taking DepthAnything3 [31] as an example, it can jointly process multiple images captured from different viewpoints of the same scene and natively output corresponding confidence maps.

Based on the above conditional information, we construct an image-to-LiDAR point cloud generation framework guided by dual conditional branches with decoupled geometry and semantics. In the first branch, we construct a geometry-conditioned encoder integrated with an uncertainty modulation mechanism to fuse raw RGB images, depth maps, and uncertainty features. In the second branch, we build a pose-aware semantic encoder to provide global semantic context and capture cross-view relationships for the model. The overall pipeline of LiFi is illustrated in Figure 1.

3.3 Reverse Sampling Based Geometry-Aware Conditioning

In this section, we describe the construction of the geometry-aware conditional branch in detail. This branch is designed to extract geometric features of the scene from multi-view camera observations, thereby providing explicit spatial structural constraints for the subsequent diffusion process. To effectively encode the geometric details contained in multi-view camera images, we first design a range-view-centered reverse depth sampling algorithm.

3.3.1 Range-View-Centered Reverse Sampling

To fully exploit the estimated depth maps D obtained from camera images, a straightforward approach is to project these depth maps pixel by pixel into LiDAR point clouds and then convert them into range images. However, this strategy fails to ensure accurate alignment between projected depth maps and LiDAR scenes, while also introducing substantial computational overhead. This issue arises because a single depth map contains millions of pixels (a common resolution is 1600×900), whereas a typical LiDAR scene is inherently sparse. Direct projection is therefore not only inefficient, but also inconsistent with the distribution pattern of LiDAR data. To address this, we introduce a range-view-centered reverse depth sampling algorithm.

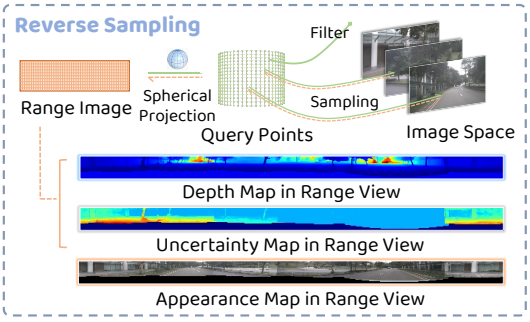


Figure 3: Schematic illustration of the reverse sampling mechanism.

Specifically, to construct the relative depth map I_D in the range-image representation, we first apply the inverse spherical projection transformation to a standard single-channel range image $I \in \mathbb{R}^{H_r \times W_r}$, yielding a set of canonical 3D query points $Q_D = \{q_k\}$. We then project these query points, which follow the distribution characteristics of LiDAR scenes, into the camera coordinate systems of different views through a spatial coordinate transformation:

$$[\tilde{u}_n, \tilde{v}_n, \tilde{w}_n]^T = K_n \cdot (R_n \cdot q_k + t_n). \tag{3}$$

Here, K_n denotes the camera intrinsic matrix of the n-th image, while R_n and t_n are the rotation and translation parameters of the corresponding camera extrinsics, respectively. We then project them into the pixel space of the camera images via perspective projection to sample the corresponding depth and uncertainty values:

$$u_n = \frac{\tilde{u}_n}{\tilde{w}_n}, \quad v_n = \frac{\tilde{v}_n}{\tilde{w}_n}. \tag{4}$$

During this process, each canonical query point, together with the LiDAR center, defines a LiDAR ray with a specific direction in 3D space. For each camera image, this ray intersects at most one pixel.

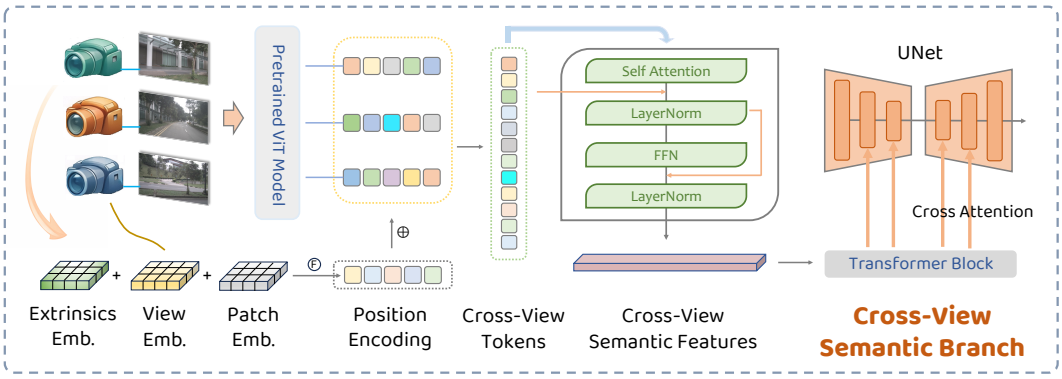


Figure 4: **Cross-View Semantic Branch** first aggregates multi-view camera tokens augmented with position encodings, then extracts cross-view features through a semantic encoder with a self-attention mechanism, and finally injects them into the latent diffusion process via Transformer blocks.

We project only valid query points that fall within the cameras' fields of view and image boundaries. Query points that cannot be projected onto any camera image are assigned null values. In cases of overlap or occlusion, we retain the depth and uncertainty values according to a nearest-depth criterion. Similarly, this algorithm can also be used to sample color information in the original RGB space. After min-max normalization, we obtain a LiDAR-aligned relative depth map I_D, an uncertainty map I_U, and an RGB image I_C.

3.3.2 Geometry-Aware Condition Encoding with Uncertainty Modulation

To effectively exploit these LiDAR-aligned visual priors, we design a geometry-aware conditional encoder. Specifically, we first introduce a 2D coordinate encoding for the range-view space, and concatenate it with the depth map and RGB image as the input to the encoder. Next, we use an encoder $\mathcal{E}_G$ to jointly model the input features. This encoder consists of an initial convolutional layer, several residual blocks, and downsampling layers, which progressively extract local geometric and appearance information and map them to appropriate feature scales. Notably, rather than simply treating the uncertainty map I_U as an additional input channel, we inject it into the encoding process as an explicit modulation signal through FiLM:

$$\tilde{H}_l = \gamma_l(I_U) \odot H_l + \beta_l(I_U), \tag{5}$$

where H_l denotes the output feature of the l-th residual block, and γ_l and β_l are the scaling and shifting terms generated by the FiLM module, respectively. This mechanism enables the model to dynamically adjust the expressive strength of the conditional features according to the confidence of local depth predictions. As a result, the encoder can preserve explicit geometric structure while suppressing the propagation of unreliable conditional information in low-confidence regions.

This branch provides local, structured, and reliable geometric constraints for LiDAR scene generation. Compared with directly using raw projection results, this design more effectively integrates depth, appearance, and spatial positional information, while enhancing the consistency between the cross-modal conditions and the target LiDAR scene.

3.4 Cross-View Semantic Conditioning

While the geometry-aware branch provides reliable geometric constraints, it is limited in capturing long-range dependencies and cross-view consistency. To address this, we introduce a cross-view semantic branch to provide scene-level semantics and contextual information. Specifically, we design a cross-view semantic encoder $\mathcal{E}_S$, which takes the flattened tokens from all views as input. We further introduce three types of positional encodings to enhance its awareness of patch coordinates, camera identity, and camera extrinsic parameters, thereby obtaining enriched feature representations:

$$\tilde{X}_S = X_S + E_{patch} + E_{view} + E_{ext}. \tag{6}$$

Here, E_{patch} denotes a set of learnable parameters of size $NL \times D$, where N is the number of cameras, L is the number of image patches per camera, and D is the feature dimension. E_{view} is

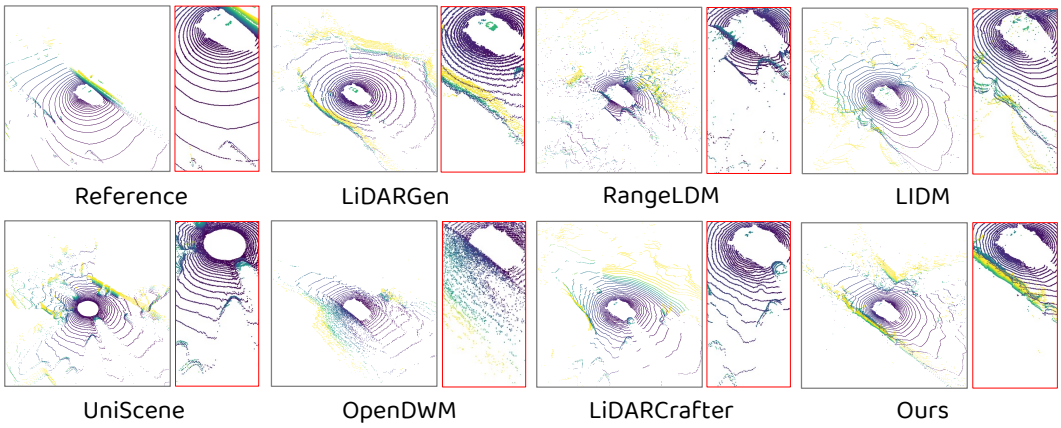


Figure 5: Visualization results of LiFi and several baseline methods for LiDAR scene generation. It can be observed that our method generates details that are closest to the real LiDAR point cloud data.

a learnable embedding vector associated with each camera index. E_{ext} is obtained by flattening the extrinsic matrix of each camera into a 16-dimensional vector and passing it through a two-layer MLP. We then feed it into a cross-view interaction module based on multi-head attention. Through self-attention, this module establishes global dependencies among tokens from all views, allowing semantic information from different camera perspectives to interact within a unified feature space:

$$H_S = \text{LayerNorm}(\tilde{X}_S + \text{SelfAttn}(\tilde{X}_S)). \tag{7}$$

Subsequently, H_S is passed through a feed-forward network and then injected into the Transformer blocks of the UNet, where it interacts with the backbone features via cross-attention. This branch further captures long-range scene relationships and cross-view consistency, thereby improving the plausibility of the generated results in terms of both global structure and semantic coherence.

3.5 Dual-Branch Balanced Classifier-Free Guidance

In the previous two sections, we introduced the geometric conditional branch and the cross-view semantic branch, respectively. However, in practice, the model may become biased toward one conditional branch during training, resulting in under-utilization of the other branch.

To mitigate this issue, we construct a dual-branch balanced classifier-free guidance (CFG) module. Standard CFG typically enhances conditional guidance at inference time by randomly dropping all conditions, thereby allowing the model to learn both conditional and unconditional distributions. In contrast, we extend the condition dropout mechanism beyond the conventional "all-or-nothing" scheme by additionally introducing cases where only geometric or only semantic conditions are retained. This enables the model not only to possess the unconditional modeling capability, but also to explicitly learn the individual effects of the two types of conditions as well as their joint contributions. During inference, we adopt the same conditional guidance formulation as in standard CFG:

$$\epsilon = \epsilon_{\text{uncond}} + w \cdot (\epsilon_{\text{cond}} - \epsilon_{\text{uncond}}), \tag{8}$$

where w is the guidance scale, ϵ_{uncond} and ϵ_{cond} denote the noise predictions under unconditional and conditional guidance, respectively.

This strategy preserves the advantages of standard CFG while further balancing the learning process between the geometric and semantic branches, enabling the model to leverage global semantic information to generate more plausible details when the depth information from the geometric branch is unreliable. With all the above designs, our model is able to jointly leverage local geometric conditions and global semantic conditions, thereby establishing reliable cross-modal correspondences between multi-view images and LiDAR scenes and generating high-fidelity LiDAR scenes.

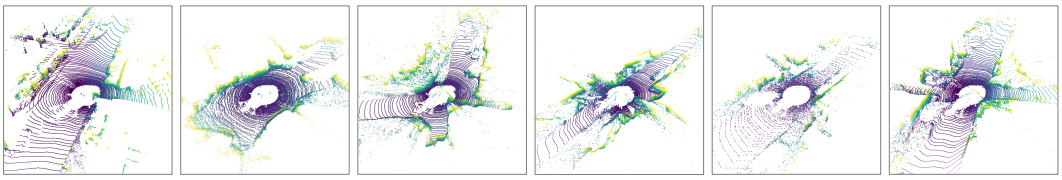


Figure 6: Qualitative results of LiFi and several major baseline models on the KITTI-360 dataset.

4 Experiments

4.1 Experimental Setups

Dataset. We conduct experiments on the challenging nuScenes dataset [2]. It contains over a thousand autonomous driving sequences collected in two geographically distinct urban environments with complex traffic conditions. The nuScenes training set includes 297,737 LiDAR scans, while the test set contains 52,423 LiDAR scans, each captured by a 32-beam LiDAR sensor. This dataset is widely used in various autonomous driving tasks, such as 3D object detection [17, 25–27], multi-object tracking [13, 38, 54], trajectory prediction [9, 29], and end-to-end autonomous driving [15, 16, 21]. We also conduct experiments on the KITTI-360 dataset [7], as detailed in the supplementary material.

Baselines. We compare LiFi with multiple categories of baseline methods. The first category includes conventional LiDAR data generation methods, covering both unconditional and conditional generation models. The second category consists of baselines capable of generating LiDAR from camera images, including LiDM [40] and X-Drive [51]. The third category includes models for generating continuous LiDAR scenes. In addition, we apply LiFi to a downstream 3D object detection task and compare the performance of models with and without LiDAR input.

Implementation Details. We conduct distributed training on four NVIDIA 141GB GPUs. Our model is trained for 200 epochs, with a total of 176,000 optimization steps, and the learning rate is set to 1×10^{-4}. We use the DDIM sampler [44] for denoising, and set the total number of diffusion timesteps T to 1024. The UNet consists of three downsampling blocks and three upsampling blocks, each containing two Transformer blocks. We employ DepthAnything3 [31] as the pretrained visual encoder, which can predict spatially consistent geometry from an arbitrary number of input views. It is an instantiation choice that matches the requirements, rather than an irreplaceable component of the framework. During inference, we perform 50 denoising steps to generate the target range image. More experimental settings and implementation details are provided in the supplementary material.

4.2 LiDAR Data Generation

This subsection evaluates the performance of LiFi on the task of generating high-fidelity LiDAR point clouds. We compare our method against both unconditional and conditional generation baselines. Following the evaluation protocol of LiDARGen [57], we adopt MMD, JSD, and two perception-based Fréchet distances as evaluation metrics. As shown in Table 1, our method achieved clear improvements across multiple metrics.

Table 1: The main quantitative results of LiFi and several baseline methods on the nuScenes dataset. The MMD values are reported in 10^{-5} and JSD in 10^{-2}.

	Method	Venue	MMD ↓	JSD ↓	FRD ↓	FPD ↓
Uncond.	LiDARGen [57]	ECCV'22	55.14	27.69	–	–
	RangeLDM [14]	ECCV'24	72.01	8.43	1172.06	441.16
	LiDM [40]	CVPR'24	4.57	10.61	2358.15	420.36
Cond.	UniScene [24]	CVPR'25	107.18	12.35	1636.60	230.89
	OpenDWM [34]	CVPR'25	10.75	7.69	2607.81	146.30
	LiDARCrafter [28]	AAAI'26	24.69	7.37	1183.11	**36.17**
	LiFi	–	**2.78**	**3.66**	**906.71**	126.31

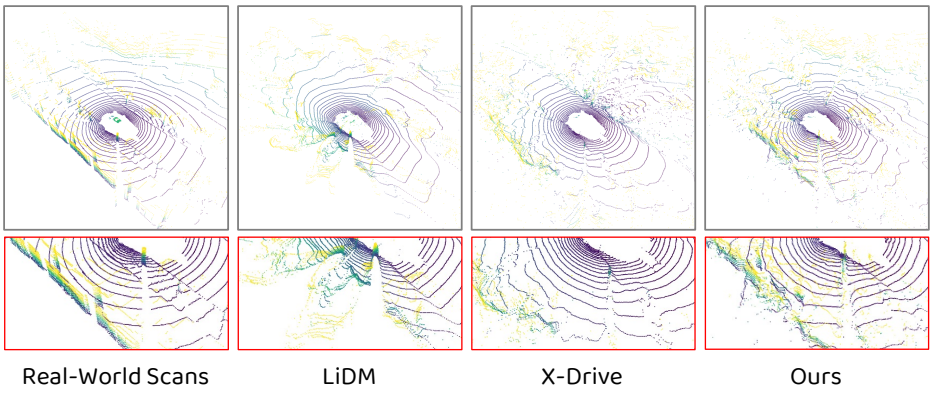


Figure 7: Qualitative results of LiDAR scene generation from camera images. The first row shows the full scenes, and the second row presents zoomed-in local details.

Table 2: The main quantitative results of LiFi and several baseline methods on the KITTI-360 dataset. The MMD values are reported in 10^{-4} and JSD in 10^{-2}.

Method	Venue	MMD ↓	JSD ↓	FRD ↓	FPD ↓
LiDARGen [57]	ECCV'22	13.36	13.67	2415.23	102.80
RangeLDM [14]	ECCV'24	4.31	9.42	1362.25	68.22
LiDM [40]	CVPR'24	9.31	19.00	2894.65	253.91
OpenDWM [34]	CVPR'25	12.93	14.63	1966.36	436.17
LiFi	–	**1.99**	**8.77**	**1107.21**	**21.40**

4.3 Camera-to-LiDAR Generation

This subsection compares LiFi with several baselines capable of generating LiDAR point clouds from camera images. We additionally introduce the CD and EMD metrics, which enable a more rigorous evaluation of point-level fidelity in the generated scenes, to assess the model's performance in generating geometrically consistent LiDAR scenes. We also introduce the DAS metric [51] to evaluate the cross-modal consistency. As shown in Table 3, LiFi achieves substantially better point-level fidelity and cross-modal consistency than existing camera-to-LiDAR baselines.

We further evaluate the quality of LiDAR data generated by LiFi across different depth ranges to investigate how its generation performance varies with scene distance and to analyze the associated failure modes. Detailed results are provided in the supplementary material.

Table 3: Quantitative results on the nuScenes dataset. These baseline models are used for generating LiDAR scenes from camera images. The MMD values are reported in 10^{-5} and JSD in 10^{-2}. Inference time is evaluated in s/frame. Evaluation is conducted on a single NVIDIA 141GB GPU.

Method	Venue	MMD ↓	JSD ↓	CD ↓	EMD ↓	DAS ↓	Time
LiDM [40]	CVPR'24	8.02	9.97	4.57	99.71	2.17	**0.34**
X-Drive [51]	ICLR'25	48.58	10.54	4.73	179.62	1.52	1.80
LiFi	–	**2.78**	**3.66**	**0.83**	**50.08**	**1.02**	0.52

4.4 Temporal Consistency of Continuous Scene Generation

We further evaluate the temporal consistency of LiDAR scenes generated by LiFi under consecutive camera inputs. While this approach does not explicitly incorporate mechanisms for temporal generation, inputting consecutive camera frames nevertheless produces LiDAR sequences that maintain good temporal consistency. Following LiDARCrafter [28], we evaluate these methods using CTC and TTCE, two metrics designed to measure temporal consistency. The results are shown in Table 4.

Table 4: Quantitative results of LiFi for continuous LiDAR scene generation. Here, TTCE and CTC are adopted to evaluate the temporal consistency of the generated results.

Method	Venue	TTCE		CTC↓				
		3	4	1	2	3	4	10
UniScene [24]	CVPR'25	2.74	3.69	0.90	1.84	3.64	3.90	–
OpenDWM [34]	CVPR'25	2.68	3.65	1.02	2.02	3.37	5.05	–
OpenDWM-DiT [34]	CVPR'25	2.71	3.66	0.89	1.79	3.06	4.64	–
LiDARCrafter [28]	AAAI'26	**2.65**	**3.56**	1.12	2.38	3.02	4.81	–
LiFi	–	2.67	3.61	**0.80**	**1.11**	**1.52**	**2.22**	**3.19**

4.5 Downstream Task: 3D Object Detection

To assess the practical value of LiFi in real-world applications, we apply it to the downstream 3D object detection task in autonomous driving. We retrained the detector using the generated LiDAR data and compared the camera-only (BEVDet [17]) and camera-LiDAR fusion baselines (BEVFusion [30]). The experimental results in Table 5 demonstrate that LiDAR point clouds generated by LiFi have certain value in downstream tasks of autonomous driving.

Table 5: Quantitative results of applying LiFi to the downstream 3D object detection task. We compare the performance of the camera-only model and the fusion-based model.

Method	mAP ↑	NDS ↑
BEVDet (Camera Only) [17]	0.278	0.341
BEVFusion (+Real LiDAR) [30]	0.529	0.533
BEVFusion (+Gen LiDAR) [30]	**0.526**	**0.520**

4.6 Ablation Study

In this subsection, we conduct ablation studies to evaluate the contribution of each component in the LiFi framework. We first conduct ablation studies on the two branches. We further perform ablations on the depth map, the original RGB image, and the uncertainty modulation mechanism in the geometric branch. The results are shown in Table 6. Finally, we perform ablation studies on the CFG module, including the dual-branch balanced design and the guidance scale parameter. The corresponding results are presented in Table 7. Furthermore, we conduct ablation studies on the number of input image views, the resolution of input range images, and the number of denoising steps during inference. Detailed results are provided in the supplementary material.

Table 6: Ablation results on the two branches and the main conditions in the LiFi framework. The MMD is reported in 10^{-5} and JSD in 10^{-2}.

LiFi	MMD ↓	JSD ↓
w/o Geometry-Aware Branch	5.20	4.27
w/o Cross-View Semantic Branch	3.57	3.67
w/o Depth Map Condition	3.57	4.12
w/o RGB Image Condition	2.80	3.92
w/o Uncertainty Modulation	2.87	3.80
w/ All	**2.78**	**3.66**

Table 7: Ablation results of the dual-branch balanced CFG module. w is the CFG scale. The MMD is reported in 10^{-5} and JSD in 10^{-2}.

LiFi	MMD ↓	JSD ↓
w/o Classifier-Free Guidance	6.32	4.85
w/o Dual-Branch Balanced	3.02	4.31
Guidance scale $w = 2.0$	5.71	3.76
Guidance scale $w = 4.0$	3.45	3.81
Guidance scale $w = 8.0$	**2.37**	4.13
w/ All ($w = 6.0$)	2.78	**3.66**

5 Conclusion

We propose LiFi, a dedicated framework for generating realistic LiDAR scenes from multi-view camera images. We introduce a dual-branch conditioning strategy to guide the latent diffusion process, effectively addressing the challenges of cross-modal geometric alignment and cross-view semantic aggregation. Experimental results show that LiFi is capable of generating high-fidelity LiDAR scenes. Moreover, the LiDAR point cloud generated by it has certain value for some downstream tasks of autonomous driving. In the future, we expect LiFi to play an important role in applications such as simulation, planning, and control in various complex environments and long-tail scenarios.

Acknowledgments and Disclosure of Funding

This research was supported by the National Science and Technology Major Project (2025ZD1606203), the Shanghai Municipal Universities Industry-Academia-Research Joint Innovation Program, the National Natural Science Foundation of China (No.62272433), Anhui Provincial Natural Science Foundation (No.2508085ZD011), Key Science & Technology Project of Anhui Province (No. 202523o09050004) and the Fundamental Research Funds for the Central Universities.

References

[1] Caccia, L., Van Hoof, H., Courville, A., Pineau, J.: Deep generative modeling of lidar data. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 5034–5040. IEEE (2019)

[2] Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11621–11631 (2020)

[3] Cai, K., Liu, X., Zhou, X., Hu, H., Xiang, J., Zhang, L., Zhang, X., Zhan, K., Zhan, Y., Lang, X.: Drivelidar4d: Sequential and controllable lidar scene generation for autonomous driving. arXiv preprint arXiv:2511.13309 (2025)

[4] Fan, H., Su, H., Guibas, L.J.: A point set generation network for 3d object reconstruction from a single image. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 605–613 (2017)

[5] Fang, J., Zuo, X., Zhou, D., Jin, S., Wang, S., Zhang, L.: Lidar-aug: A general rendering-based augmentation framework for 3d object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4710–4720 (2021)

[6] Fu, C., Mertz, C., Dolan, J.M.: Lidar and monocular camera fusion: On-road depth completion for autonomous driving. In: 2019 IEEE Intelligent Transportation Systems Conference (ITSC). pp. 273–278. IEEE (2019)

[7] Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The KITTI dataset. International Journal of Robotics Research **32**(11), 1231 – 1237 (Sep 2013)

[8] Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Advances in neural information processing systems **27** (2014)

[9] Gu, J., Hu, C., Zhang, T., Chen, X., Wang, Y., Wang, Y., Zhao, H.: Vip3d: End-to-end visual trajectory prediction via 3d agent queries. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5496–5506 (2023)

[10] Guo, X., Wu, Z., Xiong, K., Xu, Z., Zhou, L., Xu, G., Xu, S., Sun, H., Wang, B., Chen, G., et al.: Genesis: Multimodal driving scene generation with spatio-temporal and cross-modal consistency. arXiv preprint arXiv:2506.07497 (2025)

[11] Hess, G., Lindström, C., Fatemi, M., Petersson, C., Svensson, L.: Splatad: Real-time lidar and camera rendering with 3d gaussian splatting for autonomous driving. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 11982–11992 (2025)

[12] Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Advances in neural information processing systems **33**, 6840–6851 (2020)

[13] Hu, H.N., Yang, Y.H., Fischer, T., Darrell, T., Yu, F., Sun, M.: Monocular quasi-dense 3d object tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence **45**, 1992–2008 (2022)

[14] Hu, Q., Zhang, Z., Hu, W.: Rangeldm: Fast realistic lidar point cloud generation. In: European Conference on Computer Vision. pp. 115–135. Springer (2024)

[15] Hu, S., Chen, L., Wu, P., Li, H., Yan, J., Tao, D.: St-p3: End-to-end vision-based autonomous driving via spatial-temporal feature learning. In: European Conference on Computer Vision. pp. 533–549. Springer (2022)

[16] Hu, Y., Yang, J., Chen, L., Li, K., Sima, C., Zhu, X., Chai, S., Du, S., Lin, T., Wang, W., et al.: Planning-oriented autonomous driving. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 17853–17862 (2023)

[17] Huang, J., Huang, G., Zhu, Z., Ye, Y., Du, D.: Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. arXiv preprint arXiv:2112.11790 (2021)

[18] Jia, X., Gao, Y., Chen, L., Yan, J., Liu, P.L., Li, H.: Driveadapter: Breaking the coupling barrier of perception and planning in end-to-end autonomous driving. In: 2023 IEEE/CVF International Conference on Computer Vision (ICCV). pp. 7919–7929 (2023). https://doi.org/10.1109/ICCV51070.2023.00731

[19] Jia, X., Wu, P., Chen, L., Xie, J., He, C., Yan, J., Li, H.: Think twice before driving: Towards scalable decoders for end-to-end autonomous driving. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 21983–21994. IEEE (2023)

[20] Jia, X., Yang, Z., Li, Q., Zhang, Z., Yan, J.: Bench2drive: Towards multi-ability benchmarking of closed-loop end-to-end autonomous driving. Advances in Neural Information Processing Systems **37**, 819–844 (2024)

[21] Jiang, B., Chen, S., Xu, Q., Liao, B., Chen, J., Zhou, H., Zhang, Q., Liu, W., Huang, C., Wang, X.: Vad: Vectorized scene representation for efficient autonomous driving. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8340–8350 (2023)

[22] Kingma, D., Salimans, T., Poole, B., Ho, J.: Variational diffusion models. Advances in neural information processing systems **34**, 21696–21707 (2021)

[23] Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)

[24] Li, B., Guo, J., Liu, H., Zou, Y., Ding, Y., Chen, X., Zhu, H., Tan, F., Zhang, C., Wang, T., et al.: Uniscene: Unified occupancy-centric driving scene generation. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 11971–11981 (2025)

[25] Li, Y., Yu, Z., Philion, J., Anandkumar, A., Fidler, S., Jia, J., Alvarez, J.: End-to-end 3d tracking with decoupled queries. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 18302–18311 (2023)

[26] Li, Y., Ge, Z., Yu, G., Yang, J., Wang, Z., Shi, Y., Sun, J., Li, Z.: Bevdepth: Acquisition of reliable depth for multi-view 3d object detection. In: Proceedings of the AAAI conference on artificial intelligence. vol. 37, pp. 1477–1485 (2023)

[27] Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Yu, Q., Dai, J.: Bevformer: learning bird's-eye-view representation from lidar-camera via spatiotemporal transformers. IEEE Transactions on Pattern Analysis and Machine Intelligence (2024)

[28] Liang, A., Liu, Y., Yang, Y., Lu, D., Li, L., Kong, L., Zhao, H., Ooi, W.T.: Lidarcrafter: Dynamic 4d world modeling from lidar sequences. arXiv preprint arXiv:2508.03692 (2025)

[29] Liang, M., Yang, B., Zeng, W., Chen, Y., Hu, R., Casas, S., Urtasun, R.: Pnpnet: End-to-end perception and prediction with tracking in the loop. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11553–11562 (2020)

[30] Liang, T., Xie, H., Yu, K., Xia, Z., Lin, Z., Wang, Y., Tang, T., Wang, B., Tang, Z.: Bevfusion: A simple and robust lidar-camera fusion framework. Advances in neural information processing systems **35**, 10421–10434 (2022)

[31] Lin, H., Chen, S., Liew, J., Chen, D.Y., Li, Z., Shi, G., Feng, J., Kang, B.: Depth anything 3: Recovering the visual space from any views. arXiv preprint arXiv:2511.10647 (2025)

[32] Liu, Z., Tang, H., Amini, A., Yang, X., Mao, H., Rus, D., Han, S.: Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. In: IEEE International Conference on Robotics and Automation (ICRA) (2023)

[33] Nakashima, K., Kurazume, R.: Lidar data synthesis with denoising diffusion probabilistic models. In: 2024 IEEE International Conference on Robotics and Automation (ICRA). pp. 14724–14731. IEEE (2024)

[34] Ni, J., Guo, Y., Liu, Y., Chen, R., Lu, L., Wu, Z.: Maskgwm: A generalizable driving world model with video mask reconstruction. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 22381–22391 (2025)

[35] Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., Chen, M.: Glide: Towards photorealistic image generation and editing with text-guided diffusion models. arXiv preprint arXiv:2112.10741 (2021)

[36] Nichol, A.Q., Dhariwal, P.: Improved denoising diffusion probabilistic models. In: International conference on machine learning. pp. 8162–8171. PMLR (2021)

[37] Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., et al.: Dinov2: Learning robust visual features without supervision. arXiv preprint arXiv:2304.07193 (2023)

[38] Pang, Z., Li, Z., Wang, N.: Simpletrack: Understanding and rethinking 3d multi-object tracking. In: European Conference on Computer Vision. pp. 680–696. Springer (2022)

[39] Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M.: Hierarchical text-conditional image generation with clip latents. arXiv preprint arXiv:2204.06125 **1**(2), 3 (2022)

[40] Ran, H., Guizilini, V., Wang, Y.: Towards realistic scene generation with lidar diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14738–14748 (2024)

[41] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695 (2022)

[42] Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E.L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al.: Photorealistic text-to-image diffusion models with deep language understanding. Advances in neural information processing systems **35**, 36479–36494 (2022)

[43] Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. In: International conference on machine learning. pp. 2256–2265. PMLR (2015)

[44] Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. arXiv preprint arXiv:2010.02502 (2020)

[45] Song, Y., Ermon, S.: Generative modeling by estimating gradients of the data distribution. Advances in neural information processing systems **32** (2019)

[46] Song, Y., Ermon, S.: Improved techniques for training score-based generative models. Advances in neural information processing systems **33**, 12438–12448 (2020)

[47] Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. arXiv preprint arXiv:2011.13456 (2020)

[48] Van Den Oord, A., Kalchbrenner, N., Kavukcuoglu, K.: Pixel recurrent neural networks. In: International conference on machine learning. pp. 1747–1756. PMLR (2016)

[49] Wu, Y., Zhang, K., Qian, J., Xie, J., Yang, J.: Text2lidar: Text-guided lidar point cloud generation via equirectangular transformer. In: European Conference on Computer Vision. pp. 291–310. Springer (2024)

[50] Wu, Z., Ni, J., Wang, X., Guo, Y., Chen, R., Lu, L., Dai, J., Xiong, Y.: Holodrive: Holistic 2d-3d multi-modal street scene generation for autonomous driving. arXiv preprint arXiv:2412.01407 (2024)

[51] Xie, Y., Xu, C., Peng, C., Zhao, S., Ho, N., Pham, A.T., Ding, M., Tomizuka, M., Zhan, W.: X-drive: Cross-modality consistent multi-sensor data synthesis for driving scenarios. arXiv preprint arXiv:2411.01123 (2024)

[52] Xiong, Y., Ma, W.C., Wang, J., Urtasun, R.: Ultralidar: Learning compact representations for lidar completion and generation. arXiv preprint arXiv:2311.01448 (2023)

[53] Zhang, L., Rao, A., Agrawala, M.: Adding conditional control to text-to-image diffusion models. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 3836–3847 (2023)

[54] Zhang, T., Chen, X., Wang, Y., Wang, Y., Zhao, H.: Mutr3d: A multi-camera tracking framework via 3d-to-2d queries. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4537–4546 (2022)

[55] Zhou, S., Jia, X., Zhang, F., Li, J., Zhang, J., Feng, Y., Sun, J., Wong, S., You, J., Yan, J.: Lagen: Towards autoregressive lidar scene generation. arXiv preprint arXiv:2511.21256 (2025)

[56] Zhou, W., Yan, X., Liao, Y., Lin, Y., Huang, J., Zhao, G., Cui, S., Li, Z.: Bev@ dc: Bird's-eye view assisted training for depth completion. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9233–9242 (2023)

[57] Zyrianov, V., Zhu, X., Wang, S.: Learning to generate realistic lidar point clouds. In: European Conference on Computer Vision. pp. 17–35. Springer (2022)

A Overview of the Appendix

The appendix is structured into the following sections to supplement the main paper. The first part provides a discussion on the limitations of this research and directions for future work. The second part presents additional experimental details. The third part provides an in-depth theoretical analysis of the proposed modules. The fourth part supplements the experimental results.

B Limitations and Future Work

Our work presents several limitations. Primarily, relying on the range image representation is not entirely lossless, resulting in the omission of certain camera image details within the synthesized LiDAR point clouds. Refining this representation in future research holds the potential to further boost model performance. Furthermore, the overall architecture of our method is not specifically designed for explicit temporal modeling. Although this work does not focus on generating temporal LiDAR sequences, incorporating temporal consistency mechanisms in future work could extend the applicability of our method to a broader range of scenarios.

C More Implementation Details

During training, our model is trained for 200 epochs with a total of 176,000 optimization steps. The total batch size is set to 32, and the learning rate is set to 1×10^{-4}, with a cosine annealing scheduler applied. We perform distributed training on 4 NVIDIA 141GB GPUs, with a total training time of approximately 20 hours. For denoising, we employ the DDIM sampler [44], with the total number of diffusion timesteps T set to 1024. During inference, we perform 50 denoising steps to generate point clouds. The inference is conducted on a single NVIDIA 141GB GPU.

We adopt DepthAnything3 [31] as the pre-trained visual encoder. For the semantic branch, we follow DepthAnything3 and employ DINOv2 [37] to extract visual features. Based on the nuScenes dataset, we use six camera images from different viewpoints as input to generate each scene.

In the geometry-aware branch, the encoder primarily consists of multiple convolutional layers, normalization layers, downsampling layers, zero-convolution layers, and FiLM modules. In the cross-view semantic branch, the encoder mainly includes a multi-head self-attention layer, a feed-forward network (composed of two fully connected layers), and two Layer Normalization layers. Both the attention layer and the FFN employ residual connections.

The VAE in the LDM is initialized with pre-trained weights from RangeLDM [14]. The range image resolution is set to [1024, 32], and the VAE uses a downsampling factor of 4. The UNet consists of three downsampling blocks (the latter two incorporating Transformer layers) and three upsampling blocks (the first two incorporating Transformer layers). We employ circular convolution instead of standard convolution in both the VAE and the LDM.

D Detailed Description of Evaluation Metrics

In the main text, we introduced the Maximum Mean Discrepancy (MMD), Jensen-Shannon Divergence (JSD), Fréchet Point cloud Distance (FPD), and Fréchet Range Image Distance (FRD) metrics to evaluate the performance of the generator. Below, we provide a detailed explanation.

Maximum Mean Discrepancy (MMD) is a non-parametric method for measuring the distance between two probability distributions. The fundamental idea is to compute the distance between the kernel means of two sample sets in a high-dimensional space; by comparing the means mapped to this space, one can determine whether the sample distributions are consistent:

$$MMD = \frac{1}{N^2}\sum_{i}^{N}\sum_{j}^{N} k(p_i, p_j) + \frac{1}{N^2}\sum_{i}^{N}\sum_{j}^{N} k(q_i, q_j) - \frac{2}{N^2}\sum_{i}^{N}\sum_{j}^{N} k(p_i, q_j). \tag{9}$$

Jensen-Shannon Divergence (JSD) is a symmetric information-theoretic measure for quantifying the similarity between two probability distributions, derived from the Kullback-Leibler (KL) divergence

and commonly used for distribution comparison. The essence of JSD is to evaluate the information loss of each distribution with respect to their uniform mixture:

$$JSD = \frac{1}{2}\sum_{i}^{N}(p_i \log \frac{p_i}{m_i} + q_i \log \frac{q_i}{m_i}). \tag{10}$$

We compute the MMD and JSD using 160×100 2D histograms on the bird's-eye view (BEV) plane.

Fréchet Point cloud Distance (FPD) is used to measure the distance between the generated point cloud distribution and the real point cloud distribution. It utilizes a pre-trained point cloud feature extraction network (such as PointNet or PointNet++) to extract global features, and then computes the Fréchet distance between two Gaussian distributions:

$$FPD = \|\mu_r - \mu_g\|_2^2 + \mathrm{Tr}(\sigma_r + \sigma_g - 2\sqrt{\sigma_r \cdot \sigma_g}). \tag{11}$$

Fréchet Range Image Distance (FRD) is a variant specifically designed for LiDAR data. It first converts the point cloud into a Range Image and then computes the Fréchet distance. It uses a 2D CNN (such as ResNet) to extract features. The formula is identical to that of FPD, but the source of the features differs.

To evaluate the temporal consistency of the synthesized sequences, we follow LiDARCrafter [28] and adopt the TTCE (Translation and Transformation Consistency Error) and CTC (Chamfer Temporal Consistency) metrics.

Let $T_t^{gt} = [R_t^{gt} \mid t_t^{gt}]$ denote the ground-truth transformation from frame t to $t+1$, where R_t^{gt} represents the rotation matrix and t_t^{gt} is the translation vector. Let $T_t^{pred} = [R_t^{pred} \mid t_t^{pred}]$ be the transformation estimated from the generated point clouds via Iterative Closest Point (ICP). The TTCE is calculated as follows:

$$TTCE = \frac{1}{T-1}\sum_{t=1}^{T-1} \|t_t^{pred} - t_t^{gt}\|_F, \tag{12}$$

where $\|\cdot\|_F$ denotes the Frobenius norm and T is the total number of frames in the sequence.

Furthermore, let $\mathcal{P}_t$ and $\mathcal{P}_{t+1}$ be the generated point clouds at frames t and $t+1$, respectively. Using the ground-truth transformation T_t^{gt}, we align $\mathcal{P}_{t+1}$ to the coordinate system of frame t:

$$\tilde{\mathcal{P}}_{t+1} = \left(R_t^{gt}\right)^{-1}\left(\mathcal{P}_{t+1} - t_t^{gt}\right). \tag{13}$$

Subsequently, the Chamfer Distance (CD) between $\mathcal{P}_t$ and the aligned $\tilde{\mathcal{P}}_{t+1}$ is computed:

$$CD\left(\mathcal{P}_t, \tilde{\mathcal{P}}_{t+1}\right) = \frac{1}{\|\mathcal{P}_t\|}\sum_{x\in\mathcal{P}_t}\min_{y\in\tilde{\mathcal{P}}_{t+1}}\|x-y\|_2^2 + \frac{1}{\|\tilde{\mathcal{P}}_{t+1}\|}\sum_{y\in\tilde{\mathcal{P}}_{t+1}}\min_{x\in\mathcal{P}_t}\|y-x\|_2^2. \tag{14}$$

The final CTC score is defined as the average Chamfer Distance across all adjacent frame pairs:

$$CTC = \frac{1}{T-1}\sum_{t=1}^{T-1} CD\left(\mathcal{P}_t, \tilde{\mathcal{P}}_{t+1}\right). \tag{15}$$

Moreover, we introduce Chamfer Distance (CD) [4] and Earth Mover's Distance (EMD) to evaluate the fidelity of the generated point clouds in the camera-to-LiDAR task.

Chamfer Distance is a commonly used metric for measuring similarity between point clouds. It quantifies the "alignment" distance between two sets of points, representing the sum of the distances from each point in one set to its nearest neighbor in the other set. By aggregating these nearest neighbor distances across all points, Chamfer Distance captures the overall distribution difference:

$$CD(P_{\text{gen}}^s, P_{\text{gt}}^s) = \frac{1}{M}\sum_{x\in P_{\text{gt}}^s}\min_{y\in P_{\text{gen}}^s}\|x-y\|_2^2 + \frac{1}{N}\sum_{y\in P_{\text{gen}}^s}\min_{x\in P_{\text{gt}}^s}\|y-x\|_2^2. \tag{16}$$

Earth Mover's Distance (EMD) measures the minimum average cost required to transport the generated point cloud to the real point cloud. Compared with CD, EMD better reflects differences in the global structure and density distribution between two point clouds:

$$EMD(P,Q) = \min_{T\geq 0}\sum_{i=1}^{N}\sum_{j=1}^{M} T_{ij}\, d(p_i, q_j), \quad \text{s.t. } \sum_{j=1}^{M} T_{ij} = a_i,\ \sum_{i=1}^{N} T_{ij} = b_j. \tag{17}$$

Considering the actual requirements of autonomous driving scenarios, we measured the above indicators within the range of [-50.0, 50.0] m.

In downstream detection tasks for autonomous driving, we introduce the mean Average Precision (mAP) and nuScenes Detection Score (NDS) metrics to evaluate detection accuracy and overall perception quality.

The mAP metric measures the precision-recall performance of the detection model across different categories:

$$mAP = \frac{1}{C} \sum_{c=1}^{C} AP_c, \tag{18}$$

where AP is the area under the Precision-Recall curve:

$$AP = \sum_{n} (R_n - R_{n-1}) \cdot P_{interp}(R_n), \tag{19}$$

where P and R represent precision and recall, respectively.

The NDS summarizes overall 3D detection performance:

$$NDS = \frac{1}{10} \left(5 \cdot mAP + \sum_{k \in ATE, ASE, AOE, AVE, AAE} (1 - \min(1, e_k)) \right), \tag{20}$$

where ATE, ASE, AOE, AVE, and AAE represent position error, size error, orientation error, velocity error, and attribute error, respectively.

E Standard Range-View Projection and Corrected Version

Range images offer a compact representation of LiDAR data by parametrizing unstructured point clouds in 3D space into a 2D pixel space. This projection process is reversible, and after applying it, both training and inference costs for the model can be significantly reduced. Consequently, representing LiDAR data with range images has become one of the prevailing trends in the field.

The standard range image mapping method projects each point $p = (x, y, z) \in \mathbb{R}^3$ in the point cloud into the spherical coordinates (r, θ, ϕ) of the range image using spherical coordinate projection:

$$r = \sqrt{x^2 + y^2 + z^2}, \quad \theta = \arctan(y, x), \quad \phi = \arctan\left(z, \sqrt{x^2 + y^2 + z^2}\right), \tag{21}$$

where r is the depth value, θ is the azimuth angle, and ϕ is the elevation angle. RangeLDM [14] points out that multiple lasers in a LiDAR system do not share the same origin during measurement, which can introduce errors during coordinate transformation and lead to incorrect range-view data distributions. Therefore, we adopt its corrected range image representation as follows:

$$r = \sqrt{x^2 + y^2 + (z - h_j)^2}, \quad \theta = \arctan(y, x), \quad \phi = \phi_j, \tag{22}$$

where h_j and ϕ_j represent the sensor height and elevation angle estimated by Hough voting, respectively. In subsequent computations, both projection methods are processed in exactly the same way, including converting spherical coordinates (r, θ, ϕ) to range image pixel coordinates (u, v):

$$u = W - ((\theta + \pi)/2\pi) \cdot W, \quad v = H - j, \tag{23}$$

as well as normalizing the depth values of the point cloud to obtain the range image I.

In the subsequent calculation of the CD metric, it is necessary to reverse-convert the range image back into a point cloud. In the corrected version, the standard LiDAR's fifth dimension, which is the beam index j for each point, is directly used during the forward projection process. However, the generated point clouds do not contain the beam index dimension. Therefore, we introduce the following formula to compute the beam index for each point:

$$j = (\text{Rad2Deg}(\arcsin(\frac{z}{d})) - fov_{\text{down}})/(fov_{\text{up}} - fov_{\text{down}}) \cdot (H - 1), \tag{24}$$

where fov_{up} and fov_{down} refer to the elevation angle range of the original LiDAR acquisition.

F Theoretical Analysis of the Geometry-Aware Mechanism

F.1 Computational Complexity

Let the resolution of the high-resolution camera depth map be $H_{img} \times W_{img}$, and the target LiDAR range image resolution be $H_r \times W_r$. Conventional projection methods require iterating over each pixel of the depth map, calculating the 3D coordinates, and projecting them onto a cylindrical coordinate system, followed by rasterization. The computational complexity of this process is $\mathcal{O}(H_{img} \cdot W_{img})$. Since this process involves mapping from a high-resolution image to a low-resolution LiDAR grid, it typically results in many-to-one write conflicts. To address the occlusion problem, additional aggregation operations are required, which introduce extra computational overhead.

In contrast, the reverse sampling method starts directly from the target LiDAR grid. For each query point q_k, the projection via the camera is:

$$[\tilde{u}_n, \tilde{v}_n, \tilde{w}_n]^T = K_n(R_n \cdot q_k + t_n), \tag{25}$$

where K_n, R_n, and t_n are the camera's intrinsic matrix, rotation matrix, and translation vector. The overall time complexity is strictly constrained to $\mathcal{O}(H_r \cdot W_r)$. Since in physical scenes, $H_r \cdot W_r \ll H_{img} \cdot W_{img}$, the reverse sampling significantly reduces the computational overhead.

F.2 Spatial Sampling Distribution

The forward projection can be formalized as a nonlinear mapping from the image plane $\Omega_{\text{img}} \subset \mathbb{R}^2$ to the three-dimensional space:

$$\mathcal{F}(u, v) = D(u, v) \cdot K^{-1}[u, v, 1]^\top, \tag{26}$$

where $D(u, v)$ is the depth at pixel (u, v), and K^{-1} is the inverse of the camera's intrinsic matrix. Due to the nonlinearity of perspective projection, the Jacobian determinant satisfies:

$$\left| \det \frac{\partial(x, y)}{\partial(X, Y)} \right| \propto \frac{1}{Z^2}, \tag{27}$$

where Z is the depth. Therefore, uniform sampling on the image induces a depth-dependent non-uniform distribution in 3D space:

$$\rho(X, Y, Z) \propto \frac{1}{Z^2}. \tag{28}$$

This results in over-sampling in nearby regions and severe under-sampling in distant regions, leading to significant aliasing and ghost artifacts when mapped to the regular LiDAR range image grid. In contrast, our proposed reverse sampling method directly constructs the set of query points $Q_D = \{q_k\}$ in the target LiDAR domain, with a distribution that is consistent with the LiDAR's spherical scanning pattern. Specifically, the query points are generated via spherical coordinate parametrization, which is equivalent to performing structured discrete sampling on the target spherical domain, rather than performing non-uniform reprojection in the image domain.

For each query point q_k, we project it onto the image plane to obtain continuous coordinates (u_n, v_n), and then use nearest-neighbor sampling to extract the corresponding value from the depth map:

$$\hat{D}(q_k) = D_n(\lfloor u_n \rceil, \lfloor v_n \rceil). \tag{29}$$

This process can be viewed as a zero-order hold approximation of the continuous depth field, with the error arising from pixel quantization. Assuming the underlying depth field D^* is Lipschitz continuous, the sampling error satisfies:

$$|\hat{D}(q_k) - D^*(u_n, v_n)| \leq L \cdot \delta, \tag{30}$$

where L is the local Lipschitz constant, and δ is the pixel scale.

F.3 Weighted Estimation and Optimization Dynamics

In the multi-view geometric conditioning process, due to occlusion and depth estimation errors, the intermediate features H_l of the encoder are often corrupted by spatially correlated and unevenly distributed noise. We model this as:

$$H_l(x, y) = H_{\text{true}}(x, y) + \epsilon(x, y), \tag{31}$$

where the noise term satisfies:

$$\epsilon(x, y) \sim \mathcal{N}(0, \sigma^2(x, y)), \tag{32}$$

and the noise variance $\sigma^2(x, y)$ is monotonically related to the uncertainty map $I_U(x, y)$, i.e.,

$$\sigma^2(x, y) \propto I_U(x, y). \tag{33}$$

This model corresponds to a typical heteroscedastic noise setting, where different spatial locations exhibit varying levels of noise intensity.

In the above setup, a natural objective is to weight the features based on local noise intensity to reduce the impact of high-noise regions on the overall estimate. Consider the following weighted form:

$$\hat{H}(x, y) = w(x, y) \cdot H_l(x, y), \tag{34}$$

where the weight $w(x, y)$ should decrease as the noise variance increases. Classical statistical estimation theory indicates that, under certain distribution assumptions, a reasonable weight form is typically inversely proportional to the noise variance:

$$w(x, y) \propto \frac{1}{\sigma^2(x, y)}. \tag{35}$$

In our setup, since $\sigma^2(x, y)$ is approximately characterized by $I_U(x, y)$, the weight can be treated as a function of I_U:

$$w(x, y) = g(I_U(x, y)), \tag{36}$$

where $g(\cdot)$ is a monotonically decreasing function.

We introduce uncertainty-based FiLM modulation into the model:

$$\tilde{H}_l = \gamma_l(I_U) \odot H_l + \beta_l(I_U), \tag{37}$$

where $\gamma_l(\cdot)$ and $\beta_l(\cdot)$ are nonlinear mappings learned by the neural network. This can be understood as a learnable heteroscedastic weighted estimator, where the weight function is learned in a data-driven manner. In contrast, if the uncertainty I_U were directly used as an additional input channel to the network, the model would need to implicitly learn the above weighting relationship through high-dimensional nonlinear mappings. This approach presents challenges such as high learning difficulty, poor sample efficiency, and optimization instability.

It also has a direct impact on the backpropagation process. For the loss function $\mathcal{L}$, we have:

$$\frac{\partial \mathcal{L}}{\partial H_l} = \gamma_l(I_U) \odot \frac{\partial \mathcal{L}}{\partial \tilde{H}_l}. \tag{38}$$

As a result, the gradient magnitude is modulated by $\gamma_l(I_U)$, and when $I_U(x, y)$ is large, $\gamma_l(I_U)$ tends to decrease, causing the gradient to be attenuated. This enables the model to adaptively suppress gradient interference from unreliable regions during optimization, thereby improving training stability.

G Theoretical Analysis of the Cross-View Semantic Branch

In multi-view systems, the 2D images from different viewpoints are projections of the same scene under different camera poses, and the imaging process can be represented as a projection mapping $\pi : \mathbb{R}^3 \to \mathbb{R}^2$. Due to the lack of depth information, the 3D correspondence of any pixel (u, v) is a set of points along a ray:

$$\pi^{-1}(u, v) = \{X \mid \pi(X) = (u, v)\}, \tag{39}$$

which means that relying solely on 2D features for cross-view interaction is theoretically ill-posed.

To address this, we introduce patch encoding E_{patch}, viewpoint encoding E_{view}, and camera extrinsic parameter encoding $E_{\text{ext}}(T)$, to construct pose-conditioned features and explicitly introduce the conditional distribution into the feature space. This design reduces the feature confusion caused by projection uncertainty.

Building on this, we employ a self-attention mechanism to model the cross-view dependencies. This process can be formalized as an adaptive kernel regression:

$$H_i = \sum_j \kappa(\tilde{x}_i, \tilde{x}_j) \cdot x_j, \tag{40}$$

where the kernel function $\kappa(\cdot,\cdot)$ is implicitly defined by feature similarity. Since the features contain pose information, this kernel function is able to learn cross-view association patterns related to 3D geometric relationships during training, thereby promoting geometric consistency in the latent space. Furthermore, if we define the cross-view feature consistency error as

$$\mathcal{L}_{cons} = \sum_{i,j} A_{ij} \|x_i - x_j\|^2, \tag{41}$$

the above attention update is equivalent to a kernel smoothing operation, which can reduce this consistency error during optimization, thus achieving cross-view feature alignment.

At the same time, our method achieves the complementary integration of local and global information through the collaborative modeling of the geometric and semantic branches. Specifically, the local geometric branch tends to capture high-frequency spatial structures (such as depth discontinuities and boundaries), while the attention-based semantic branch models long-range dependencies by globally weighted aggregation, focusing more on low-frequency structures. In summary, this design can be understood as learning a pose-conditioned kernel regression operator, which preserves local details while effectively enhancing cross-view consistency through global information interaction.

H Theoretical Analysis of the Dual-Branch Balanced CFG

In conditional diffusion models, the introduction of a conditional variable $c = (c_g, c_s)$ aims to learn the conditional distribution $p(x \mid c)$, ensuring that the generated results maintain a statistical dependency with the condition variable. Although the training objective of the diffusion model does not directly maximize mutual information, the maximum likelihood objective can be interpreted as implicitly encouraging the increase of mutual information between x and the condition variables. According to the chain rule for mutual information, we have:

$$I(x; c_g, c_s) = I(x; c_g) + I(x; c_s \mid c_g), \tag{42}$$

In standard joint training, if the geometric condition c_g provides strong structural constraints, the model may preferentially utilize c_g to minimize the loss during the early stages of optimization, thereby weakening its reliance on the semantic condition c_s. This phenomenon can be interpreted as an information shortcut, where the model tends to rely on the conditioning variable that is easier to optimize. To mitigate this issue, we sample the condition c from the following set during training:

$$c \sim \{\emptyset, c_g, c_s, (c_g, c_s)\}, \tag{43}$$

and the corresponding optimization objective is defined as:

$$\mathcal{L} = \mathbb{E}_c \mathbb{E}_{x,\epsilon,t}[\|\epsilon - \epsilon_\theta(x_t, c, t)\|^2]. \tag{44}$$

This objective is equivalent to simultaneously fitting multiple conditional distributions:

$$p(x), \quad p(x \mid c_g), \quad p(x \mid c_s), \quad p(x \mid c_g, c_s). \tag{45}$$

From the information-theoretic perspective, this training strategy explicitly encourages the model to maintain robust predictive capabilities across different conditional subspaces, thereby preventing the model from relying solely on a single conditioning variable and mitigating the issue of imbalanced information utilization.

I Discussion on Pre-trained Vision Models and Metric Scale

Replaceability of the Pretrained Vision Model.

From the perspective of the framework interface, LiFi does not depend on the specific internal architecture of DepthAnything3. The geometric branch only requires the depth foundation model to provide two types of per-pixel information for each input view: depth predictions and their corresponding reliability estimates. LiFi's reverse sampling module then aligns these predictions with the target LiDAR scanning grid, while the uncertainty-aware encoder uses the reliability signals to modulate the geometric features. Therefore, as long as an alternative model can provide outputs in the same form, the subsequent inverse sampling, FiLM modulation, and diffusion generation pipeline can remain unchanged.

Therefore, from an architectural perspective, any large monocular depth model capable of estimating per-pixel uncertainty can technically be directly integrated into LiFi. However, if the model processes each camera view independently, the predicted depths may have independent scales across different views. In this case, we recommend introducing cross-view scale alignment before reverse sampling.

Specifically, one feasible approach is to use the known camera calibration and overlapping fields of view to perform scale-and-shift alignment across the depth maps from different views. For example, the prediction for each view can be expressed as: $\hat{D_n} = a_n D_n + b_n$, where a_n and b_n are estimated by enforcing consistency within the overlapping regions. After cross-view alignment, a unified normalization scheme can be applied to all views before feeding them into LiFi.

Therefore, multi-view consistency is not a strict architectural requirement for LiFi, but it can positively contribute to maintaining the quality of the geometric conditions. In addition, methods with multi-view depth estimation capabilities, such as VGGT and MapAnything, can serve as direct replacements for DepthAnything3. Therefore, DepthAnything3 is an instantiation choice that matches the task requirements, rather than an irreplaceable component of the LiFi framework.

Metric Scale of the Simulator.

We first emphasize that the final LiDAR generated by LiFi is already represented in the metric LiDAR coordinate system of the target dataset, rather than having only relative scale.

LiFi trains the diffusion model with range images constructed from real LiDAR scans as the target, and therefore learns a mapping from relative visual geometry and image semantics to real metric LiDAR distance distributions. After de-normalization, each valid cell in the generated range image represents an actual distance in the target sensor coordinate system. It can then be back-projected using the corresponding laser-beam elevation angles, azimuth sampling, and sensor calibration parameters to obtain a 3D point cloud in meters. The range image transformation adopted in the paper is itself reversible, and the generation target is constructed from the metric distances of real LiDAR scans.

Therefore, for datasets such as nuScenes and KITTI-360, there is no need to fit an additional global scale factor after generation. Metric LiDAR point clouds from datasets such as KITTI and Argoverse can likewise be directly converted into range images according to the scanning pattern of the target sensor and used as training supervision for LiFi.

If LiFi is to be adapted to a specific simulator, we recommend the following procedure:

1. If the simulator is intended to use a target LiDAR configuration consistent with those of current mainstream datasets, the LiFi architecture trained on these datasets can be directly used.

2. If the goal is to simulate a completely new virtual target LiDAR configuration, the first step is to specify the target LiDAR settings, including the beam elevation angles, horizontal angular resolution, and extrinsic parameters. Based on these settings, the target range image representation can then be constructed, and the VAE and diffusion model can be trained or fine-tuned using camera–metric LiDAR data corresponding to that sensor. At inference time, the generated results can be converted back into a metric point cloud according to the same normalization parameters and sensor model.

J Analysis of Computational Efficiency

In this section, we provide an analysis of the inference efficiency of LiFi. As shown in Table 8, we present the performance and inference speed of our method across different denoising timesteps.

K Ablation Study on the Number of Input Image Views

The number of input RGB images does not affect the fairness of the experimental comparisons but is merely an architectural design choice. In this section, we present an ablation study on the nuScenes dataset using 1, 2, 4, and 6 input camera images, as shown in Table 9.

Table 8: Quantitative evaluation of LiFi's inference efficiency. The MMD values are reported in 10^{-5} and JSD in 10^{-2}. Inference time is evaluated in s/frame.

Denoising Step	MMD	JSD	CD	Time
$T_{\text{inference}} = 10$	180.97	18.00	1.29	0.11
$T_{\text{inference}} = 30$	0.98	4.29	1.00	0.31
$T_{\text{inference}} = 50$	2.78	3.66	0.83	0.52
$T_{\text{inference}} = 100$	2.73	3.84	0.83	1.04
$T_{\text{inference}} = 200$	2.76	3.83	0.82	2.08

Table 9: Quantitative evaluation of the impact of the number of input images on LiFi's generation performance. The MMD values are reported in 10^{-5} and JSD in 10^{-2}.

Num.	MMD ↓	JSD ↓	CD ↓	EMD ↓
1	14.98	4.83	1.93	112.54
2	7.46	4.35	1.53	99.48
4	5.85	4.17	1.19	76.34
6	2.78	3.66	0.83	50.08

L Ablation Study on the Range Image Resolution

In this section, we conduct an ablation study on the resolution of the range images, with the results presented in Table 10. At a resolution of $[1024, 32]$, most pixels correspond to LiDAR points in the scene, whereas $[512, 32]$ results in the omission of approximately half of the points in the complete LiDAR scene during representation conversion. In contrast, $[2048, 32]$ introduces nearly an equal number of additional empty pixels, which can instead interfere with learning in valid regions.

Table 10: Quantitative evaluation of LiFi's generation performance and inference efficiency across different range image resolutions. The MMD values are reported in 10^{-5} and JSD in 10^{-2}.

Range Image Size	MMD ↓	JSD ↓	Time ↓
$[512, 32]$	740.77	31.04	0.29
$[1024, 32]$	2.78	3.66	0.52
$[2048, 32]$	3.32	4.03	0.97

M Quantitative Analysis of Failure Cases at Long Range

In this section, we evaluate the generation performance of our model across different depth ranges. As shown in Table 11, LiFi's performance gradually degrades as the distance increases.

Table 11: Quantitative evaluation of LiFi's generation performance across different radial distance ranges. The MMD values are reported in 10^{-5} and JSD in 10^{-2}.

Radial Distance	Proportion of Points	MMD ↓	JSD ↓	CD ↓	EMD ↓
$[0, 10)$ m	60.87%	2.08	2.06	0.11	2.05
$[10, 20)$ m	23.69%	1.40	3.80	0.94	34.62
$[20, 30)$ m	9.25%	2.48	5.52	4.13	189.64
$[30, 40)$ m	3.92%	5.18	6.61	12.06	538.58
$[40, +\infty)$ m	2.27%	7.03	9.23	32.82	1264.71

N More Qualitative Results

In this section, we first present additional visual examples comparing the generation results of our method with those of LiDAR generation baselines, as shown in Figure 8. We then provide more camera-to-LiDAR visualization results across diverse scenarios, as shown in Figure 9.

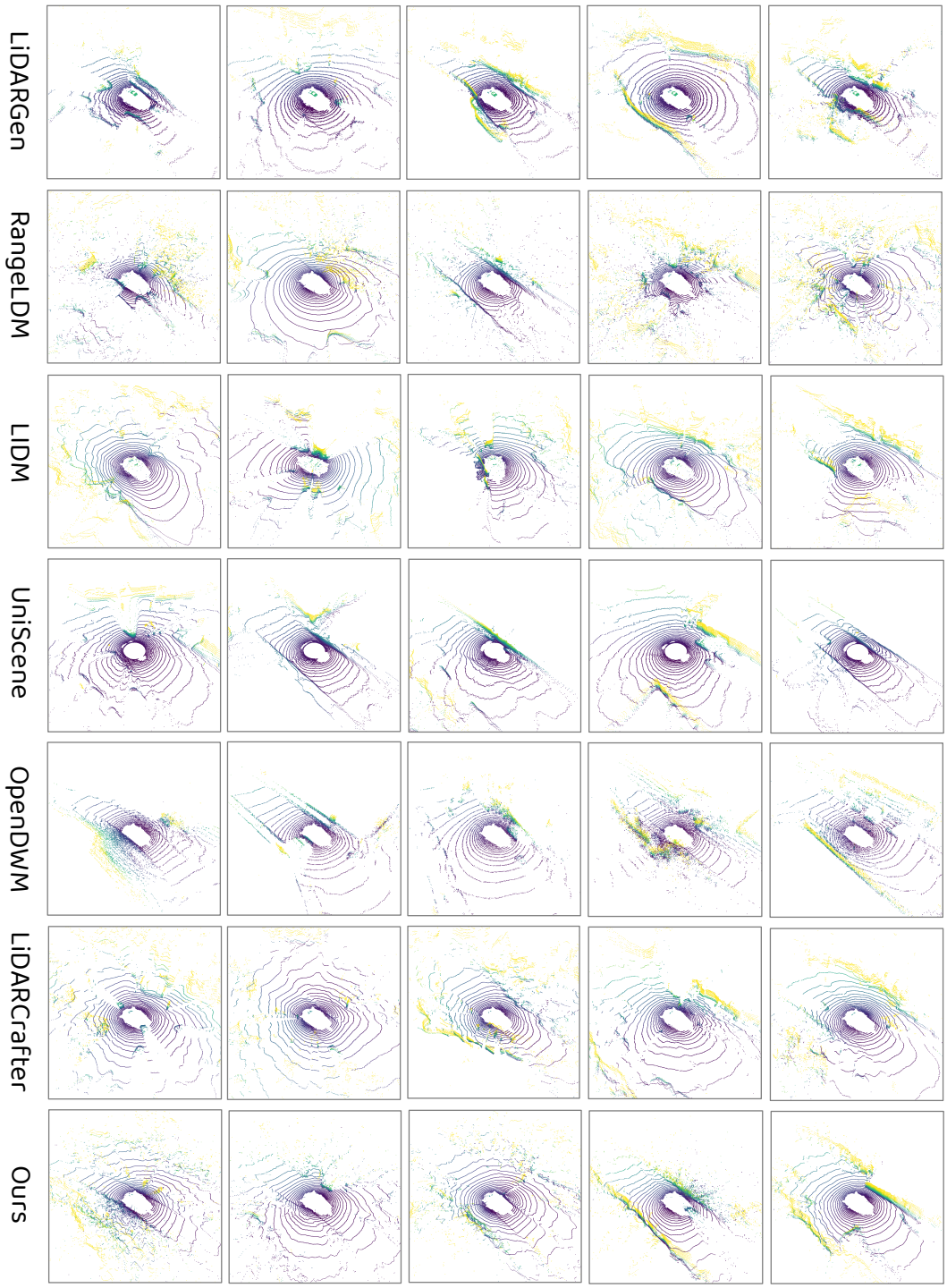


Figure 8: Additional qualitative generation examples of our method and several baseline approaches on the LiDAR generation task on the nuScenes dataset.

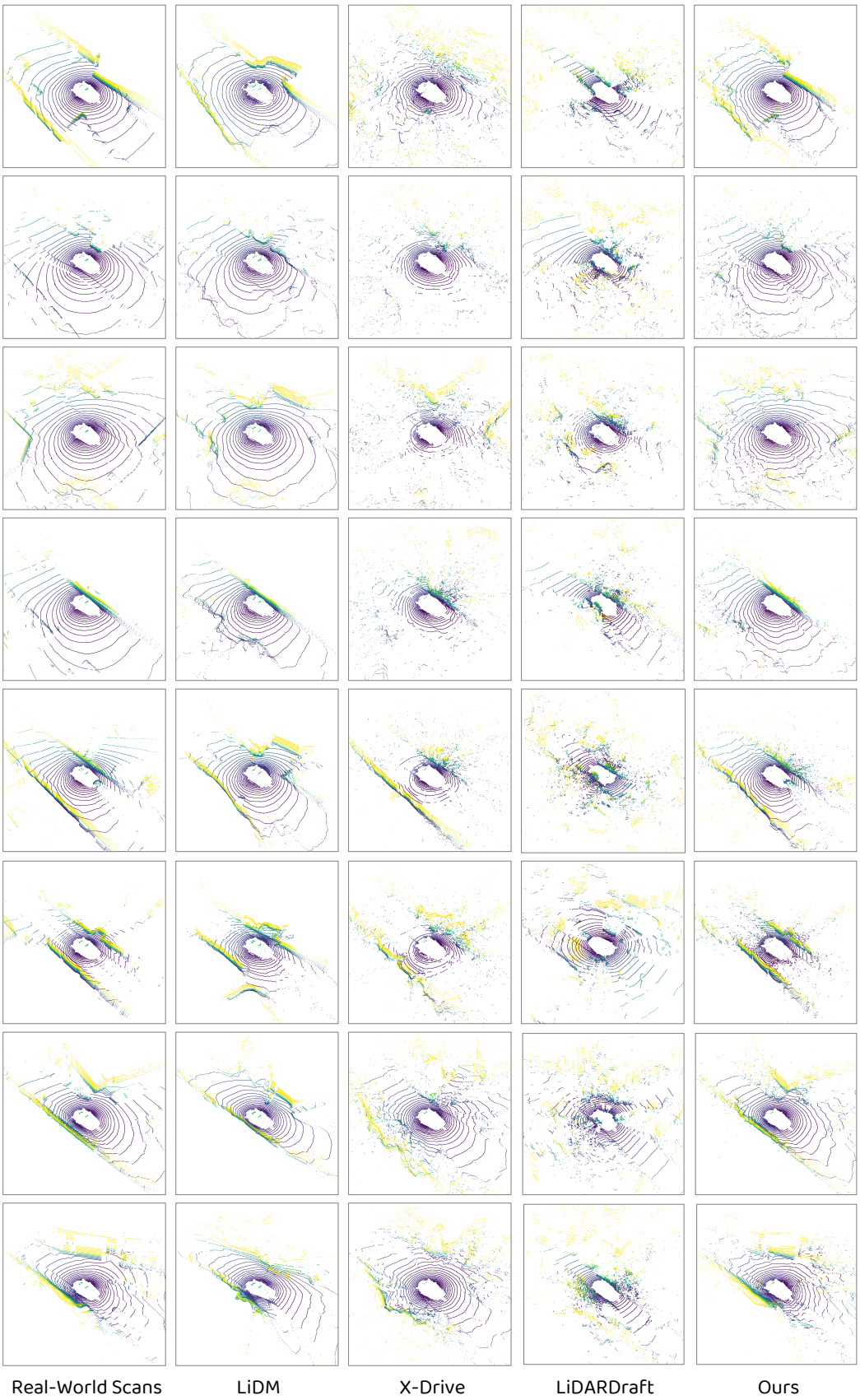


Figure 9: Qualitative comparisons of our method with several baseline approaches on the camera-to-LiDAR task across more representative scenarios on the nuScenes dataset.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflect the contributions and scope of this paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses the limitations of the work performed by the authors.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: All the theorems, formulas, and proofs in the paper have been numbered and cross-referenced, and all assumptions have been clearly stated or referenced in the statement of any theorems.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper has fully disclosed all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and conclusions of the paper.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We will open access to the code after the paper is accepted.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We have specified all the training and test details necessary to understand the results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: This paper does not include error bars or formal statistical significance tests.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper provides sufficient information on the computer resource for experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conforms in every respect with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: This paper discusses the potential social impacts of the work performed.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [No]

    Justification: We do not foresee any high risk for misuse of this work.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The creators or original owners of assets used in the paper are properly credited, and the license and terms of use are explicitly mentioned and properly respected.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in the paper does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.