LidarDM: Generative LiDAR Simulation in a Generated World

Vlas Zyrianov[1*] Henry Che[1*] Zhijian Liu[2] Shenlong Wang[1]

Abstract— **We present LidarDM, a novel LiDAR generative model capable of producing *realistic*, *layout-aware*, *physically plausible*, and *temporally coherent* LiDAR videos. LidarDM stands out with two unprecedented capabilities in LiDAR generative modeling: (i) LiDAR generation guided by driving scenarios, offering significant potential for autonomous driving simulations, and (ii) 4D LiDAR point cloud generation, enabling the creation of realistic and temporally coherent sequences. At the heart of our model is a novel integrated 4D world generation framework. Specifically, we employ latent diffusion models to generate the 3D scene, combine it with dynamic actors to form the underlying 4D world, and subsequently produce realistic sensory observations within this virtual environment. Our experiments indicate that our approach outperforms competing algorithms in realism, temporal coherency, and layout consistency. We additionally show that LidarDM can be used as a generative world model simulator for training and testing perception models. We release our source code and checkpoints at `https://github.com/vzyrianov/LidarDM`**

I. INTRODUCTION

Generative models are notable in areas such as image and video generation [1], [2], [3], [4], 3D generation [5], [6], [7], compression [8], [9], and editing [10], [11]. More recently, they have shown great promise in robotics by generating realistic scenarios and sensory data for training and validating embodied agents, significantly reducing the cost of real-world experiments [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23].

While advancements in conditional image and video generation [24], [25], [26], [27] have been remarkable, the specific task of generatively creating scenario-specific, realistic LiDAR point cloud sequences for autonomous driving application remains under-explored. Current LiDAR generation methods fall into two broad categories, each of which suffers from specific challenges. (i) *LiDAR generative modeling methods* [28], [29], [30], [31] are currently limited to single-frame generation and do not provide the means for semantic controllability and temporal consistency. (ii) *LiDAR resimulation* [32], [33], [34], [13], [35], [36] relies on user-created or real-world collected assets, this induces a high cost, restricts diversity, and limits broader applicability.

To address these challenges, we propose LidarDM (Lidar Diffusion Model), which creates *realistic*, *layout-aware*, *physically plausible*, and *temporally coherent* LiDAR videos. LidarDM enables two previously unexplored capabilities : (i) controllable LiDAR synthesis guided by driving scenarios, which holds immense potential for simulation in autonomous driving, and (ii) 4D LiDAR synthesis for generating realistic and temporally coherent sequences of labeled LiDAR point clouds. Our key insight lies in first generating and composing the underlying 4D world and then creating realistic sensory observations within this virtual environment. We develop a novel approach for large-scale 3D scene generation based on the latent diffusion model and integrate existing 3D object generators for dynamic actors. This method produces realistic and diverse 3D driving scenes from a coarse semantic layout, which to our knowledge, is one of the first of its kind. Finally, we simulate dynamic and realistic driving scenario, compose the 3D world at each time step, and perform stochastic raycasting to produce the final 4D LiDAR sequence. Our generated results are diverse, realistic, temporally coherent, and align with the layout (Fig. 1).

Our experimental results demonstrate that individual frames generated by LidarDM exhibit *realism* and *diversity*, with performance on-par with state-of-the-art techniques in unconditional single-frame LiDAR point cloud generation. Moreover, we show that LidarDM can produce *temporally coherent* LiDAR videos, outperforming a robust stable diffusion baseline. We further demonstrate LidarDM's *conditional generation* by showing that the generated LiDAR matches well with ground-truth LiDAR on matching map conditions for both geometry and intensity. Among generative LiDAR methods, LidarDM is the first to support map-conditioned generation and the first to support sequence generation. Lastly, we illustrate that the data generated by LidarDM exhibit a *minimal domain gap* when tested with perception modules trained on real data and can also be used to augment training data to significantly improve 3D detectors and planners. This gives premise for using generative LiDAR models to create realistic and controllable simulations for training and testing driving models. For detailed applications of LidarDM, please refer to Sec. II and Fig. 2.

II. RELATED WORKS

1) LiDAR Simulation: Realistic LiDAR sensor simulation is crucial for robotics and self-driving vehicle training and testing. Simulators like CARLA [32] and AirSim [37] create environments with static (buildings, trees, street lights) and dynamic (cars, bicycles, buses) assets and simulate LiDAR with raycasting. Such approaches are simple and easy to integrate, hence are widely used in robot simulation [38], [39]. However, these methods face limitations in realism and scalability due to two key issues: (i) the need for 3D

This project is supported by NSF Awards #2331878, #2340254, and #2312102 and Intel, IBM, Amazon, NCSA, and Meta.

* Both authors contribute equally to the research.

[1] V. Zyrianov, H. Che, S. Wang are with UIUC. Email: {vlasz2, hungdc2, shenlong}@illinois.edu

[2] Z. Liu is with NVIDIA. Email: zhijianl@nvidia.com

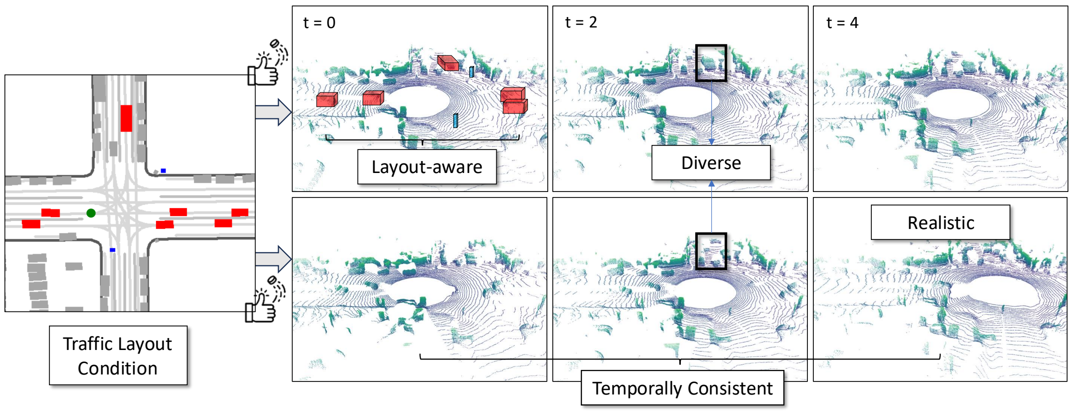

Fig. 1: LiDAR sequences generated by LidarDM are realistic, layout-conditioned, physically plausible, and temporally coherent. We show 3 frames of 2 different videos generated from the same layout. In the layout visualization, the red rectangles are dynamic vehicles, gray rectangles are static vehicles, blue dots are pedestrians, the green dot is the ego vehicle sensor, and the gray curves are road boundaries and lane markings.).

assets, which are costly and limit variations; (ii) the sim2real gap for both asset design and physics simulation.

Recent data-driven approaches reconstruct objects and environments from real-world LiDAR as meshes [13], [40] or neural fields [41], [36] from which LiDAR are simulated from raycasting or volume rendering. However, these methods require rendering LiDAR from models constructed from real-world scenes, which is costly and not scalable. In contrast, LidarDM creates scenes in a purely generative manner, eliminating the need for man-made or reconstructed assets and environments, and can generate point clouds from environments unseen during training. For example, Champs-Élysées (in Fig. 2 (a)) was created from a hand-crafted map layout, which no re-simulation method can achieve.

2) LiDAR Generation: Generative models provide a promising alternative for creating realistic LiDAR point clouds without reconstructing real-world environments. Early LiDAR generation works utilized the range image representation using GANs [30], VAEs [30], and diffusion models [28], [42], [43], [44]. Later works used voxel representations with VQGANs [29]. However, these methods only generate single frame LiDAR, and do not provide controllable or video generation. LidarDM addresses these issues by conditionally generating a 4D world and performing physics-based raycasting. As shown in Fig.2 (b) and (c), LidarDM provides realistic LiDAR data to traffic simulators thanks to its temporal consistency and can train perception models (Sec.IV-E) with paired semantic layout and generated LiDAR—benefits unmatched by other generative methods.

III. Layout-Guided LiDAR Video Generation

Our goal is to create a realistic, physically plausible, and temporally consistent LiDAR sequence that enables a free viewpoint based on a given bird's eye view semantic layout in a purely generative manner without relying on any pre-collected assets like 3D maps. The key to achieving this lies in first generating and composing the underlying 3D world, followed by using generative simulation to create realistic sensory observations. We begin by formulating generation as a joint 4D scene generation task (Sec. III-A). Next, we discuss leveraging 3D diffusion models to create static and dynamic elements, and ensuring faithful interactions (Sec. III-B). Finally, a sensor generation procedure is executed to produce the final LiDAR video (Sec. III-C). Fig. 3 depicts the overview of our method.

A. Problem Formulation

Formally, given an input layout $\mathcal{I} \in \mathbb{R}^{L\times W\times M}$ representing traffic elements from a bird's eye view where L, W, and M are length, width, and map classes (lanes, roads, crosswalks), respectively, our goal is to generate a LiDAR point cloud video $\mathcal{X} = \{\mathbf{x}_t\}$, with each $\mathbf{x}_t \in \mathbb{R}^{N\times 4}$ being a point cloud (3D location and intensity) at frame t with $\mathbf{x}_0$ matching the input layout. This conditional generation setting offers full controllability, and hence lays the foundation for a practical asset-free simulator. Without a map, our approach defaults to unconditional generation (i.e., in Sec. IV-C).

1) 4D World Representation: Our key technical innovation to address the challenge lies in jointly modeling the generation of underlying 4D world together with sensor generation. We define the world scene representation as $\mathcal{W} = \{\mathbf{s}, \{\mathbf{o}_i\}_{i=0}^N\}$, where $\mathbf{s}$ represents a static scene geometry and $\mathbf{o}_0, ..., \mathbf{o}_N$ are dynamic objects. Both are represented in the form of an occupancy grid. To model dynamics, we additionally consider the actions of these dynamic objects in the form of trajectories $\mathcal{P} = \{\boldsymbol{\tau}_0, ..., \boldsymbol{\tau}_T\}$, with $\boldsymbol{\tau}_t = \{\boldsymbol{\xi}_{\text{ego}}, \{\boldsymbol{\xi}_{i,t}\}_{i=0}^N\}$ representing the pose of actor i at time t as well as egocar pose $\boldsymbol{\xi}_{\text{ego}}$. The pose for rigid objects and the egocar lies in the $\mathbb{SE}(3)$ space, while for articulated objects like pedestrians, it is represented as a kinematic chain. A composed scene represent the states of the world at t, incorporating the poses of the ego car and dynamic objects at time t, is denoted by $\mathcal{W}_t = \pi(\mathcal{W}, \boldsymbol{\tau}_t)$, where π is an operator composing actors to world.

2) 4D World and LiDAR Generation: To ensure realism and consistency over time and between the world and sensory readings, we formulate the generation task as a sampling problem from the joint distribution $p(\mathcal{X}, \mathcal{P}, \mathcal{W}|\mathcal{I})$. Directly

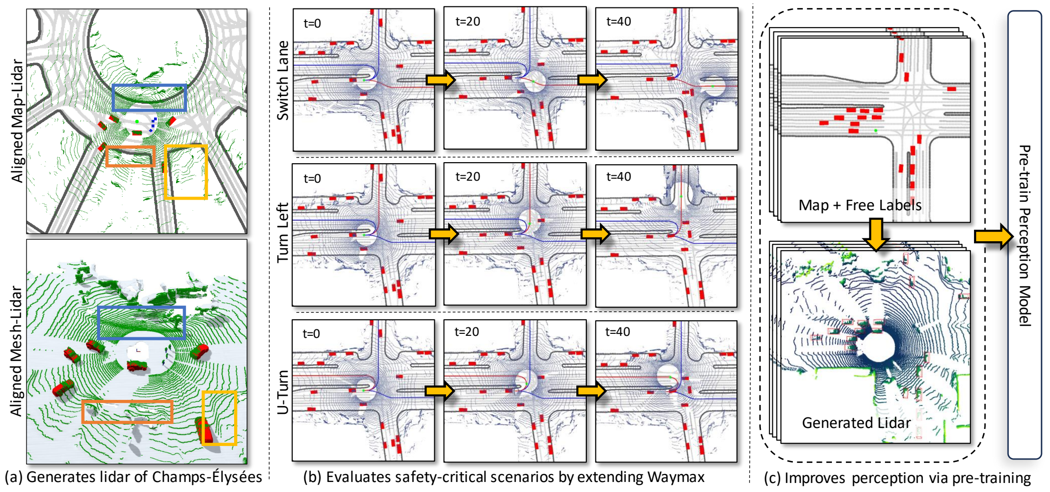
Aligned Map-Lidar
Aligned Mesh-Lidar
(a) Generates lidar of Champs-Élysées
Switch Lane
Turn Left
U-Turn
t=0
t=20
t=40
(b) Evaluates safety-critical scenarios by extending Waymax
Map + Free Labels
Generated Lidar
Pre-train Perception Model
(c) Improves perception via pre-training

Fig. 2: Applications of LiDarDM: (a) generating layout-conditioned LiDAR (color boxes highlight the lidar-map consistency); (b) creating sensor data for traffic simulator [45] to enable end-to-end safety-critical scenarios evaluation; (c) generate large volume Lidar data with known ground truth labels to improve perception models without data capturing and labelling.

modeling and sampling the joint distribution, however, is challenging as it involves estimating a distribution across multiple data modalities (e.g., car trajectories, scene layouts, sensor noise, etc.). To tackle this, we factorize the joint distribution $p(\mathcal{X}, \mathcal{P}, \mathcal{W}|\mathcal{I})$ as follows:

$$\underbrace{p(\mathbf{s}|\mathcal{I}) \cdot \prod_i p(\mathbf{o}_i|\mathcal{I})}_{\text{3D scene and object gen}} \cdot \underbrace{\prod_t p(\boldsymbol{\tau}_t|\boldsymbol{\tau}_{<t}, \mathcal{W}, \mathcal{I})}_{\text{trajectory gen}} \cdot \underbrace{\prod_t p(\mathbf{x}_t|\boldsymbol{\tau}_t, \mathcal{W})}_{\text{sensor simulation}}.$$

Next, we will discuss each individual task in detail.

B. Scene, Object and Trajectory Generation

We decompose the world into a static, intensity-infused background scene that is constant over time and dynamic foreground objects that move. This decomposition simplifies the challenging 4D world generation into manageable tasks: creating object geometries and generating dynamic effects, while ensuring temporal consistency (*e.g.* constant cars' shapes and walls and trees over time) and physical plausibility (*e.g.* ensuring correct collision reasoning).

1) Scene Generation: The scene generation step addresses the problem of sampling the geometry and LiDAR intensity of a scene from a given input layout $\mathcal{I}$: $\mathbf{s} \sim p(\mathbf{s}|\mathcal{I})$. We parameterize the intensity-infused scene $\mathbf{s} \in \mathbb{R}^{2\times L\times W\times H}$, where each entry $s_j \in \mathbb{R}^2$ encodes the truncated signed distance to the surface and an intensity value.

Model: We leverage the latent diffusion model [2], [2] to tackle this challenge of modeling and sampling from $p(\mathbf{s}|\mathcal{I})$ due to its capacity to sample high-quality data while effectively incorporating strong conditional guidance. Specifically, our model encodes the high-dimensional $\mathbf{s}$ into a continuous latent representation $\mathbf{z}$ using an encoder-decoder structure [2] with a scene encoder $E_\theta(\mathbf{s}) = \mathbf{z}$ and a scene decoder $D_\theta(\mathbf{z}) = \tilde{\mathbf{s}}$. This encoder-decoder structure efficiently compresses the input data into a lower-dimensional latent space, enabling more effective and efficient sampling. Additionally, we encode our high-definition map layout $\mathcal{I}$ into a latent space $\mathbf{c} = M_\theta(\mathcal{I})$, allowing for more compact conditioning.

Sampling: We leverage a probabilistic denoising diffusion model [2], [46] $F_\theta(\mathbf{z}, \mathbf{c})$ to perform classifier-free guidance sampling [47]. Specifically for each diffusion step k, the following Langevin dynamics step is performed to progressively denoise until a clean sample $\mathbf{z}_0$ is acquired:

$$\mathbf{z}_{k-1} = \mathbf{z}_k + \frac{\lambda_k}{2}\left[(1+w)F_\theta(\mathbf{z}_k, \mathbf{c}) - wF_\theta(\mathbf{z}_k)\right] + \sqrt{\lambda_k}\boldsymbol{\epsilon}_k$$

$F_\theta(\mathbf{z}_k, \mathbf{c})$ is the score function $\nabla_{\mathbf{z}} \log p(\mathbf{z}|\mathbf{c})$ of the conditional distribution at $\mathbf{z}_k$ and $F_\theta(\mathbf{z}_k) = F_\theta(\mathbf{z}_k, \mathbf{c} = \mathbf{0})$ is the the score function of the unconditional distribution $p_\theta(\mathbf{z})$. w is the CFG guidance scale parameter, λ_k is an annealed noise schedule parameter, and $\boldsymbol{\epsilon}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Finally, a 3D scene sample $\mathbf{s}$ is recovered by decoding the reverse-diffused sampling latent code $\mathbf{s} = E_\theta(\mathbf{z}_0)$.

Training: We train our diffusion-based scene generation model using a dataset that pairs scene geometry with map conditioning. Direct access to dense scene geometry is not available in practice. Instead, we use NKSR [48], a textured 3D reconstruction method, to recover a pseudo-GT from an input LiDAR sequence. The recovered mesh's texture encodes intensity. Ground truth annotations are used to remove moving dynamic objects, ensuring our reconstruction contains only the static scene and objects. We then train the auto-encoders for both scene geometry and map layout using reconstruction loss and KL divergence loss: $\min_\theta \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{KL}}$ over real-world examples. Our latent diffusion model is trained using the score matching loss function: $\mathcal{L}_{\text{LDM}} = \mathbb{E}_{(\mathbf{z},\mathbf{c}),\boldsymbol{\epsilon},k}\left[\|\boldsymbol{\epsilon} - F_\theta(\mathbf{z}_k, k, \mathbf{c})\|_2^2\right]$, where $\mathbf{z}_k$ is the forward diffused noisy sample at step k .

2) Object Generation: We employ two object generation frameworks, GET3D [5] and AvatarClip [49], to create dynamic traffic participants. For each actor $\mathbf{o}_i$ in a given layout $\mathcal{I}$, we sample a random variable $\mathbf{z} \sim \mathcal{N}$ and generate the corresponding actor mesh following $\mathbf{o}_i = G(\mathbf{z})$, where $G(\cdot)$ is the generator/decoder of the chosen generative method. We use GET3D to generate vehicles and then rescale them to properly fit within a target bounding box of the layout.

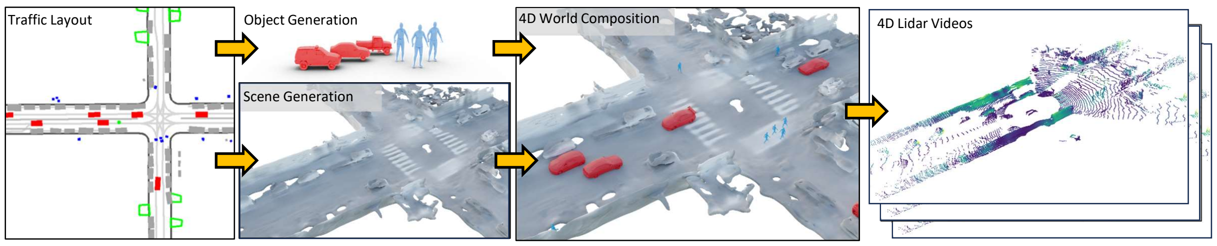

Fig. 3: Overview of the LidarDM: Given the input traffic layout at time $t = 0$, LidarDM begins by generating actors and the intensity-infused static scene. We then generate the motion of the actors and the egocar, and compose the underlying 4D world. Finally, a generative- and physics-based simulation is used to create realistic 4D sensor data.

For pedestrian generation, we utilize AvatarClip [49], which is conditioned on a SMPL [50] pose and shape parameter $\mathbf{p} = (\boldsymbol{\theta}, \boldsymbol{\beta})$. We use Mixamo [51] to animate the rigged model, ensuring realistic 4D human walking motion.

Together, the generated static world $\mathbf{s}$ and each actor $\mathbf{o}_i$ define our 3D world, denoted as $\mathcal{W}$, as depicted in Fig. 3

3) Trajectory Generation: We extend Waymax [45], a data-driven 2D BEV traffic simulator, to control the behaviors of traffic actors in more systematic manners. Given a scenario from the WOMD Dataset [52], we use Waymax to replay ego-vehicle's and agent's real-world trajectories, with an additional reactive Intelligent Driver Model [53] that updates each agent's acceleration to avoid collisions. For unconditional generation, we sample trajectories from a trajectory bank obtained from Waymo Open dataset [54]. We employ heuristics to ensure physical feasibility (no hovering) and that no collisions occur (through agent-agent or agent-scene collisions). Around **12.3%** of sampled trajectories are retained, which is acceptable because resampling is trivial.

This approach renders our world generation to be completely asset-free, end-to-end generative, and temporally consistent, allowing for a realistic and physics-based simulation without the need for artist-curated [32] or pre-collected assets [13], [35] as in previous LiDAR simulation methods.

C. Physics-Informed LiDAR Generation

Given the complete 4D world $\mathcal{W}$ and the poses $\mathcal{P}$, our next step is to generate a realistic LiDAR point cloud corresponding to these conditions. At a high level, we use the poses to compose the scene and objects at each timestep, then perform physics-informed ray casting to obtain purely physically simulated LiDAR as an intermediate result. We leverage data-driven conditional sampling to generate the final point cloud to simulate real-world LiDAR noises.

1) Scene Composition: We use Dual Marching Cube [55] to obtain the 3D mesh of the static world from the TSDF channel of the generated $\mathbf{s}$. Then, using our generated actor trajectories, we transform all 3D agents' meshes to the world coordinates and compose it with $\mathbf{s}$ using ego poses and all actors actor poses at time t, τ_t, producing the full world geometry at each time t: $\mathcal{W}_t = \pi(\mathcal{W}, \boldsymbol{\tau}_t)$. For vehicles, π applies a rigid transform. For pedestrians, π applies a rigid transform and articulates the human body shape to simulate animated movement with forward kinematics [56].

2) Physics-based Ray Casting: LiDAR sensors acquire a 3D point cloud by shooting beams of light from the sensor into the scene. Raycasting simulates this process for a single beam by calculating the $\{x, y, z\}$ coordinate point where a beam would intersect with the scene based on a given ego vehicle position τ_t, the composed scene $\mathcal{W}_t$, and the beam's elevation (θ) and azimuth (ϕ) angles. LiDARs typically have multiple beams that spin to generate a single 3D point cloud. Therefore, the process is repeated for each beam based on the virtual sensor configuration which we match with the real-world LiDARs based on provided KITTI HDL-64E [57] and Waymo specifications.

3) Intensity Modeling: For each point $\overline{\mathbf{x}}^s_{it} \in \overline{\mathbf{x}}_t$ that lies on the background scene, we query the intensity channel of the volume $\mathbf{s}$ for the intensity of the closest vertex. For intensity values of dynamic objects, we opt for a physical-based approach following Lambert's Cosine Law. Namely, for each point $\overline{\mathbf{x}}^o_{it} \in \overline{\mathbf{x}}_t$ that lies on the dynamic object, $intensity(\overline{\mathbf{x}}^o_{it}) = \alpha + \beta \left(I_{it} \cdot N^o_t\right)$ where I_{it} denotes the vector from the lidar sensor to $\overline{\mathbf{x}}^o_{it}$, N^o_t denotes the normal vector of the object at $\overline{\mathbf{x}}^o_{it}$, and α, β are constants that have been hand-tuned to match the empirical intensity distribution of dynamic objects in Waymo Dataset.

4) Stochastic Raydrop: Raycasted LiDAR from the generated world appears overly clean, without real-world environmental and sensor noise. Inspired by LiDARSim, we stochastically simulates "raydrop", where rays do not return to the sensor. For each raycast scan at time t, $\overline{\mathbf{x}}_t$, we project it onto a 2D spherical range image and predict raydrop probability per pixel on this image using a U-Net architecture [58], supervised by real-world LiDAR scan raydrop masks. Our approach, unlike LiDARSim, requires only a range map, eliminating the need for multiple additional metadata input channels that are only available in real-world data. We sample from this mask with a Gumbel sigmoid to produce the final LiDAR scan $\mathbf{x}_t$ for each frame, concluding the end-to-end LiDAR video generation process.

IV. Experiment

A. Setup

1) Datasets: We evaluate LidarDM on the KITTI-360 [59] and Waymo Open [54] datasets. KITTI-360 contains nine driving sequences (76,715 samples), where the first sequence is used as a val sequence (11,518 samples) and the last eight are used for training (65,197 samples). However,

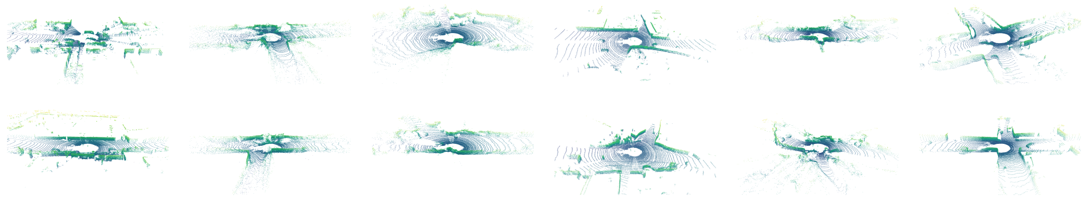

Fig. 4: Real KITTI-360 samples vs unconditional samples from the competing methods. LidarDM generates samples that feature more detailed salient objects (e.g., cars, pedestrians), sharper 3D structures (e.g., walls), and realistic road layouts.

KITTI-360 does not provide detailed BEV HD map information limiting its applications in conditional models. Waymo Open [54] is a dataset containing 1048 sequences with 158,000 training and 29,700 validation frames. The dataset provides an HD map in a vector format which we rasterize into a segmentation map and use for training the conditional model. The dimensions of the map tensor are $\mathtt{L}\times\mathtt{W}\times\mathtt{M}$ (length×width×map classes). The map has 5 classes (lane markings, road lines, edges, crosswalks, driveways).

2) Training Details: We train our models for 48 hours using four Nvidia A100 40GB GPUs. We use the Adam optimizer with a learning rate of 1e-4 for the VAE and 1e-5 for the diffusion U-Net, with a cosine decay schedule.

3) Model Details: The latent diffusion is a UNet with 5 ResNet blocks, featuring channels of 128, 128, 256, 512, 512. The SDF VAE has 4 ResNet blocks with channels of 448, 640, 896, 1280. The Map VAE has ResNet down/upsample blocks with channels of 64, 64, 128, 256, 512.

B. Baselines

1) Unconditional Generation: LiDARVAE, LiDARGAN, ProjectedGAN, LiDARGen, and R2DM are baselines that use range image representation whereas UltraLiDAR uses BEV voxel space. For fair comparison, we follow UltraLiDAR and evaluate MMD and JSD on a histogram of voxel occupancy instead of voxel density [29].[1]

2) Temporal Coherency: We are the first to attempt the task of sequential LiDAR generation and thus no previous models exist for comparison. Nonetheless, we implement a sequence diffusion baseline inspired by recent work in video generation. Concretely, we train a VAE to encode individual LiDAR frames. This has been shown previously [29] to be effective. Next, we train a diffusion model to directly denoise multiple (*i.e.*, 5) LiDAR frames at once.

C. Unconditional Single-Frame Generation

We first validate our model architecture design and showcase our model's generative capability by directly comparing against previous LiDAR generation models in unconditional generation (without using HD maps) on KITTI-360 [59]. Based on the results in Table I, BEV models (ours or UltraLiDAR) perform best compared to range image models. Note that UltraLiDAR was directly trained on the task of modeling

Method	Int.	Con.	Tem.	MMD_{BEV} (↓)	JSD_{BEV} (↓)
LiDAR VAE [30]	✗	✗	✗	8.53e−4	0.267
LiDAR GAN [30]	✗	✗	✗	8.95e−4	0.243
ProjectedGAN [60]	✗	✗	✗	7.07e−4	0.201
LidarGen [28]	✓	✗	✗	2.95e−4	0.136
UltraLidar [29]	✗	✗	✗	9.67e−5	0.132
R2DM [44]	✓	✗	✗	3.60e−4	0.148
LidarDM (Ours)	✓	✓	✓	1.67e−4	**0.119**

TABLE I: Qualitative results for unconditional generation on KITTI-360 dataset. (Int: Intensity, Con: Controllability, Tem: Temporal Consistency) (🟨 best, ⬜ second best, 🟧 third best)

Metrics	Sequence Diffusion	LidarDM
Total ICP Energy [m] (↓)	3616.58	916.94
Average ICP Energy (↓)	0.078	0.014
Outlier Percentage (↓)	20.56%	7.12%
Chamfer Distance [m] (↓)	0.39	0.17

TABLE II: Temporal consistency. Outlier percentage uses distance threshold $\tau = 0.5m$.

single LiDAR scans which the benchmark evaluates, which explains the performance gap compared to ours. We also show qualitative comparisons against the baselines in Fig. 4.

D. Map-conditioned Multi-Frame Generation

Our model is the first fully generative LiDAR model that can generate controllable (through map conditioning), realistic, and temporally coherent synthetic LiDAR scans. We will then validate these properties in this section.

1) Consistent 4D Generation: One of our key contributions is the temporal consistency of the sequential LiDAR generation. To evaluate this, we first use ICP alignment to calculate a relative transformation between consecutive generated frames. We define an **average point-to-plane energy** over a sequence of LiDAR scans as our quantitative metrics, following this equation: $E = \frac{1}{T}\sum_{t=1}^{T}\mathtt{point2plane}(x_t, x_{t-1})$ where $\mathtt{point2plane}$ represents the point-to-plane distance [61], and x_t indicates the LiDAR scan at time t. Intuitively, E is prone to higher values from dynamic objects, but it is still a valuable metric to determine if the general scene geometry is preserved over time. To further evaluate the geometric consistency , we also measure the **outlier point ratio**, defined as the percentage of points with the $\mathtt{point2plane}$ distance larger than a certain threshold τ. Table II shows our quantitative results, where we beat the baseline in both metrics by a notable margin,

[1]LiDM has random yaw rotations in their samples that can cause unfair comparison (confirmed with authors). We will include once they provide rotation-free samples

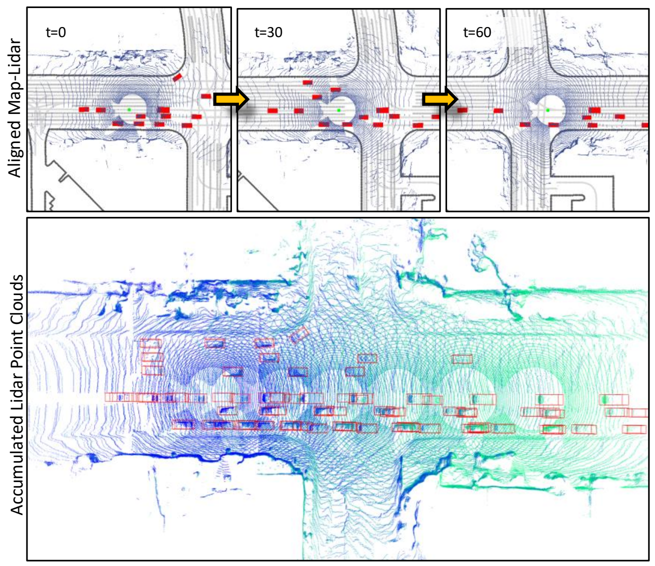

Fig. 5: Qualitative results of map-conditioned generation. Accumulated LiDAR shows temporal and layout consistency.

clearly demonstrating LidarDM's temporal consistency.

2) Layout-aware LiDAR Generation: To ascertain the layout-awareness of our LidarDM, we use CenterPoint [62] trained on real-world LiDAR scans to validate whether it can accurately detect objects from the LidarDM's LiDAR scan.

Given an input layout $\mathcal{I}$, we generate the corresponding LiDAR scan, run CenterPoint on it, and evaluate using mean average precision (mAP) for vehicles. From our experiments, LidarDM achieves comparable mAP score (56.4%) compared to real LiDAR (59.7%), indicating strong semantic correlation of LidarDM's point cloud to ground-truth. We compute the mAP agreement between our generated LiDAR scan and raw LiDAR scan to be up to 81.1%, showcasing a strong agreement between the two and demonstrating our approach's map-awareness and realism.

3) Qualitative results: We show qualitative results of our map-conditioned LiDAR sequence generation in Fig. 5. Our generated results closely match the map conditioning, and the accumulated points over 90 frames highlights LidarDM's temporal consistency and map-awareness. The use of physical-based LiDAR sensor simulation guarantees that the generated point clouds are properly occluded by obstacles and appear as a realistic LiDAR sweep pattern.

4) Intensity Evaluation: To assess our intensity realism, we use Wasserstein distance (WD) to compare the intensity distribution of our generated point cloud with that of the real-world point cloud. As a baseline, we adopt the physics-based intensity model described in Sec. III-C.3. Our experiment shows that LidarDM's intensity distribution matches well with the gt (WD of 0.0197), which significantly outperforms the physics-based baseline (WD of 0.0813).

E. Augmenting Real Data with LidarDM

LidarDM is the first LiDAR generative model capable of generating data conditioned on semantic layouts. This capability offers the potential to augment the training data for and improve 3D perception and planning models.

Config	L2 (m) @ 1.0s	L2 (m) @ 2.0s	L2 (m) @ 3.0s	Collision Rate (%)
9.2k Real	0.489	1.374	3.279	1.65%
9.2k Real+92k LidarDM	0.490	1.341	3.160	1.12%

TABLE III: **Planner data augmentation**: LidarDM-generated data can enhance performance of end-to-end planner. (🟨 indicates best)

1) Perception Model Sim2Real: We first use LidarDM to generate around 70k frames of simulation data based on the layout from Waymo Dataset [54]. After that, we pre-train a LiDAR-based 3D object detection model, CenterPoint [62] (with PointPillars [63] as its backbone), on these generated LiDAR frames, paired with the object labels from the dataset. We then train the same model on 35k frames of real data, both with and without the pre-training stage on the simulation data, to test the benefits of the LidarDM-generated data. LidarDM-augmented model achieves mAP score of 61.3%, compared to 58.2% achieved with the real data-only model. This shows that LidarDM is an effective generative data augmentation strategy, offering over 3% improvement in detection accuracy.

2) Planning Model Sim2Real: Inspired by NMP [64], we developed a learning-based motion planner that takes the five most recent LiDAR observations (covering 0.5 seconds of past history) as input and generates 10 frame (with 0.3 second intervals between each) cost map of the car's motion plan. We make two changes to the original NMP model: 1) our planner does not require privileged HD Maps as input, allowing the experimental results to focus on the quality of our generated LiDAR, and 2) the planner does not explicitly incorporate the ego car's past trajectory [65], [66], [67]. We employ a soft cross-entropy loss to train the cost map and sample from a trajectory bank (generated from the Waymo dataset using K-Means) during inference.

To show LidarDM's benefit for motion planning, we first train a model on 92k LidarDM-generated snippets, then fine-tune it on 9.2k real sequences. Expert driver trajectories serve as ground truth (GT) with traffic layout conditions generating LidarDM samples. For comparison, the same model is trained on only the 9.2k real sequences. Both models are trained for 30 epochs until convergence.

We report our results in Table III Using generative pre-training improves the performance of the planner in a low-data regime. In particular, our collision rate after 3 seconds has been reduced by 32% (relative). To our knowledge, this is the first time conditional LiDAR generation has been shown to improve an end-to-end motion planner.

V. Conclusion

We presented LidarDM, a novel layout-conditioned latent diffusion model for generating realistic LiDAR point clouds. Our approach frames the problem as a joint 4D world creation and sensory data generation task and develops a novel latent diffusion model to create 3D scenes. The resulting point cloud videos are realistic, coherent, and layout-aware.

References

[1] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *CVPR*, 2019.

[2] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *CVPR*, 2022.

[3] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, "Hierarchical text-conditional image generation with clip latents," *ArXiv*, 2022.

[4] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, S. K. S. Ghasemipour, B. K. Ayan, S. S. Mahdavi, R. G. Lopes, T. Salimans, J. Ho, D. J. Fleet, and M. Norouzi, "Photorealistic text-to-image diffusion models with deep language understanding," *ArXiv*, 2022.

[5] J. Gao, T. Shen, Z. Wang, W. Chen, K. Yin, D. Li, O. Litany, Z. Gojcic, and S. Fidler, "Get3d: A generative model of high quality 3d textured shapes learned from images," in *Advances In Neural Information Processing Systems*, 2022.

[6] B. Poole, A. Jain, J. T. Barron, and B. Mildenhall, "Dreamfusion: Text-to-3d using 2d diffusion," *arXiv*, 2022.

[7] M. Li, P. Zhou, J.-W. Liu, J. Keppo, M. Lin, S. Yan, and X. Xu, "Instant3d: Instant text-to-3d generation," *arXiv preprint arXiv:2311.08403*, 2023.

[8] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," in *ICLR*, 2017.

[9] L. Huang, S. Wang, K. Wong, J. Liu, and R. Urtasun, "Octsqueeze: Octree-structured entropy model for lidar compression," in *CVPR*, 2020.

[10] C. Meng, Y. He, Y. Song, J. Song, J. Wu, J.-Y. Zhu, and S. Ermon, "SDEdit: Guided image synthesis and editing with stochastic differential equations," in *ICLR*, 2022.

[11] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *CVPR*, 2017.

[12] X. Zhu, V. Zyrianov, Z. Liu, and S. Wang, "Mapprior: Bird's-eye view perception with generative models," 2023.

[13] S. Manivasagam, S. Wang, K. Wong, W. Zeng, M. Sazanovich, S. Tan, B. Yang, W.-C. Ma, and R. Urtasun, "Lidarsim: Realistic lidar simulation by leveraging the real world," in *CVPR*, 2020.

[14] Y. Chen, F. Rong, S. Duggal, S. Wang, X. Yan, S. Manivasagam, S. Xue, E. Yumer, and R. Urtasun, "Geosim: Realistic video simulation via geometry-aware composition for self-driving," in *CVPR*, 2021.

[15] S. Tan, K. Wong, S. Wang, S. Manivasagam, M. Ren, and R. Urtasun, "Scenegen: Learning to generate realistic traffic scenes," *CVPR*, 2021.

[16] L. Feng, Q. Li, Z. Peng, S. Tan, and B. Zhou, "Trafficgen: Learning to generate diverse and realistic traffic scenarios," in *ICRA*, 2023.

[17] S. W. Kim, , J. Philion, A. Torralba, and S. Fidler, "DriveGAN: Towards a Controllable High-Quality Neural Simulation," in *CVPR*, 2021.

[18] M. Yang, Y. Du, K. Ghasemipour, J. Tompson, D. Schuurmans, and P. Abbeel, "Learning interactive real-world simulators," *arXiv preprint arXiv:2310.06114*, 2023.

[19] J. Liang, R. Liu, E. Ozguroglu, S. Sudhakar, A. Dave, P. Tokmakov, S. Song, and C. Vondrick, "Dreamitate: Real-world visuomotor policy learning via video generation," 2024.

[20] A. Swerdlow, R. Xu, and B. Zhou, "Street-view image generation from a bird's-eye view layout," *arXiv preprint arXiv:2301.04634*, 2023.

[21] Y. Shen, B. Chandaka, Z.-h. Lin, A. Zhai, H. Cui, D. Forsyth, and S. Wang, "Sim-on-wheels: Physical world in the loop simulation for self-driving," *arXiv preprint arXiv:2306.08807*, 2023.

[22] A. Blattmann, R. Rombach, H. Ling, T. Dockhorn, S. W. Kim, S. Fidler, and K. Kreis, "Align your latents: High-resolution video synthesis with latent diffusion models," in *CVPR*, 2023.

[23] T. Yu, T. Xiao, A. Stone, J. Tompson, A. Brohan, S. Wang, J. Singh, C. Tan, J. Peralta, B. Ichter *et al.*, "Scaling robot learning with semantically imagined experience," in *RSS*, 2023.

[24] S. W. Kim, B. Brown, K. Yin, K. Kreis, K. Schwarz, D. Li, R. Rombach, A. Torralba, and S. Fidler, "Neuralfield-ldm: Scene generation with hierarchical latent diffusion models," in *CVPR*, 2023.

[25] A. Hu, L. Russell, H. Yeo, Z. Murez, G. Fedoseev, A. Kendall, J. Shotton, and G. Corrado, "Gaia-1: A generative world model for autonomous driving," 2023.

[26] S. Luo, Y. Tan, L. Huang, J. Li, and H. Zhao, "Latent consistency models: Synthesizing high-resolution images with few-step inference," *arXiv preprint arXiv:2310.04378*, 2023.

[27] P. Esser, J. Chiu, P. Atighehchian, J. Granskog, and A. Germanidis, "Structure and content-guided video synthesis with diffusion models," in *ICCV*, 2023.

[28] V. Zyrianov, X. Zhu, and S. Wang, "Learning to generate realistic lidar point cloud," in *ECCV*, 2022.

[29] Y. Xiong, W.-C. Ma, J. Wang, and R. Urtasun, "Learning compact representations for lidar completion and generation," in *CVPR*, 2023.

[30] L. Caccia, H. v. Hoof, A. Courville, and J. Pineau, "Deep generative modeling of lidar data," in *IROS*, 2019.

[31] L. Zhang, Y. Xiong, Z. Yang, S. Casas, R. Hu, and R. Urtasun, "Learning unsupervised world models for autonomous driving via discrete diffusion," *arXiv preprint arXiv:2311.01017*, 2023.

[32] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *CoRL*, 2017.

[33] A. Tallavajhula, C. Mericli, and A. Kelly, "Off-road lidar simulation with data-driven terrain primitives," in *ICRA*, 2018.

[34] J. Fang, D. Zhou, F. Yan, T. Zhao, F. Zhang, Y. Ma, L. Wang, and R. Yang, "Augmented lidar simulator for autonomous driving," *RAL*, 2020.

[35] Z. Yang, Y. Chen, J. Wang, S. Manivasagam, W.-C. Ma, A. J. Yang, and R. Urtasun, "Unisim: A neural closed-loop sensor simulator," in *CVPR*, 2023.

[36] T. Tao, L. Gao, G. Wang, Y. Lao, P. Chen, H. Zhao, D. Hao, X. Liang, M. Salzmann, and K. Yu, "Lidar-nerf: Novel lidar view synthesis via neural radiance fields," 2023.

[37] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *FSR*, 2017.

[38] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science Robotics*, 2022.

[39] A. Afzal, C. L. Goues, and C. S. Timperley, "Gzscenic: Automatic scene generation for gazebo simulator," *arXiv preprint arXiv:2104.08625*, 2021.

[40] J. Schmidt, Q. Khan, and D. Cremers, "LiDAR View Synthesis for Robust Vehicle Navigation Without Expert Labels," in *ITSC*, 2023.

[41] S. Huang, Z. Gojcic, Z. Wang, F. Williams, Y. Kasten, S. Fidler, K. Schindler, and O. Litany, "Neural lidar fields for novel view synthesis," in *ICCV*, 2023.

[42] H. Ran, V. Guizilini, and Y. Wang, "Towards realistic scene generation with lidar diffusion models," in *CVPR*, 2024.

[43] Q. Hu, Z. Zhang, and W. Hu, "Rangeldm: Fast realistic lidar point cloud generation," in *ECCV*, 2024.

[44] K. Nakashima and R. Kurazume, "Lidar data synthesis with denoising diffusion probabilistic models," in *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2024, pp. 14 724–14 731.

[45] C. Gulino, J. Fu, W. Luo, G. Tucker, E. Bronstein, Y. Lu, J. Harb, X. Pan, Y. Wang, X. Chen, J. D. Co-Reyes, R. Agarwal, R. Roelofs, Y. Lu, N. Montali, P. Mougin, Z. Yang, B. White, A. Faust, R. McAllister, D. Anguelov, and B. Sapp, "Waymax: An accelerated, data-driven simulator for large-scale autonomous driving research," in *NeurIPS*, 2023.

[46] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," in *NeurIPS*, 2019.

[47] J. Ho, "Classifier-free diffusion guidance," *ArXiv*, 2022.

[48] J. Huang, Z. Gojcic, M. Atzmon, O. Litany, S. Fidler, and F. Williams, "Neural kernel surface reconstruction," in *CVPR*, 2023.

[49] F. Hong, M. Zhang, L. Pan, Z. Cai, L. Yang, and Z. Liu, "Avatarclip: Zero-shot text-driven generation and animation of 3d avatars," *ACM Transactions on Graphics (TOG)*, vol. 41, no. 4, pp. 1–19, 2022.

[50] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "SMPL: A skinned multi-person linear model," *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, vol. 34, no. 6, pp. 248:1–248:16, Oct. 2015.

[51] Adobe Inc., "Mixamo," https://www.mixamo.com/.

[52] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. Qi, Y. Zhou, Z. Yang, A. Chouard, P. Sun, J. Ngiam, V. Vasudevan, A. McCauley, J. Shlens, and D. Anguelov, "Large scale interactive motion forecasting for autonomous driving : The waymo open motion dataset," *arXiv*, 2021.

[53] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical Review E*, vol. 62, pp. 1805–1824, 02 2000.

[54] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov, "Scalability in perception for autonomous driving: Waymo open dataset," in *CVPR*, 2020.

[55] S. Schaefer and J. Warren, "Dual marching cubes: primal contouring of dual grids," in *12th Pacific Conference on Computer Graphics and Applications, 2004. PG 2004. Proceedings.*, 2004, pp. 70–76.

[56] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black, "Keep it smpl: Automatic estimation of 3d human pose and shape from a single image," in *ECCV*, 2016.

[57] "High definitian lidar hdl-64e," https://hypertech.co.il/wp-content/uploads/2015/12/HDL-64E-Data-Sheet.pdf.

[58] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "RangeNet++: Fast and Accurate LiDAR Semantic Segmentation," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[59] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d," *PAMI*, 2022.

[60] A. Sauer, K. Chitta, J. Müller, and A. Geiger, "Projected gans converge faster," in *NeurIPS*, 2021.

[61] K.-L. Low, "Linear least-squares optimization for point-to-plane icp surface registration," 01 2004.

[62] T. Yin, X. Zhou, and P. Krähenbühl, "Center-based 3d object detection and tracking," *CVPR*, 2021.

[63] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," 2019.

[64] W. Zeng, W. Luo, S. Suo, A. Sadat, B. Yang, S. Casas, and R. Urtasun, "End-to-end interpretable neural motion planner," 2021.

[65] F. Codevilla, E. Santana, A. M. López, and A. Gaidon, "Exploring the limitations of behavior cloning for autonomous driving," *ICCV*, 2019.

[66] D. Dauner, M. Hallgarten, A. Geiger, and K. Chitta, "Parting with misconceptions about learning-based vehicle motion planning," in *CoRL*, 2023.

[67] J.-T. Zhai, Z. Feng, J. Du, Y. Mao, J.-J. Liu, Z. Tan, Y. Zhang, X. Ye, and J. Wang, "Rethinking the open-loop evaluation of end-to-end autonomous driving in nuscenes," *arXiv preprint arXiv:2305.10430*, 2023.