RadarGen: Automotive Radar Point Cloud Generation from Cameras

Tomer Borreda[1] Fangqiang Ding[1,2] Sanja Fidler[3,4,5]
Shengyu Huang[3] Or Litany[1,3]
[1] Technion [2] MIT [3] NVIDIA [4] University of Toronto [5] Vector Institute
https://radargen.github.io/

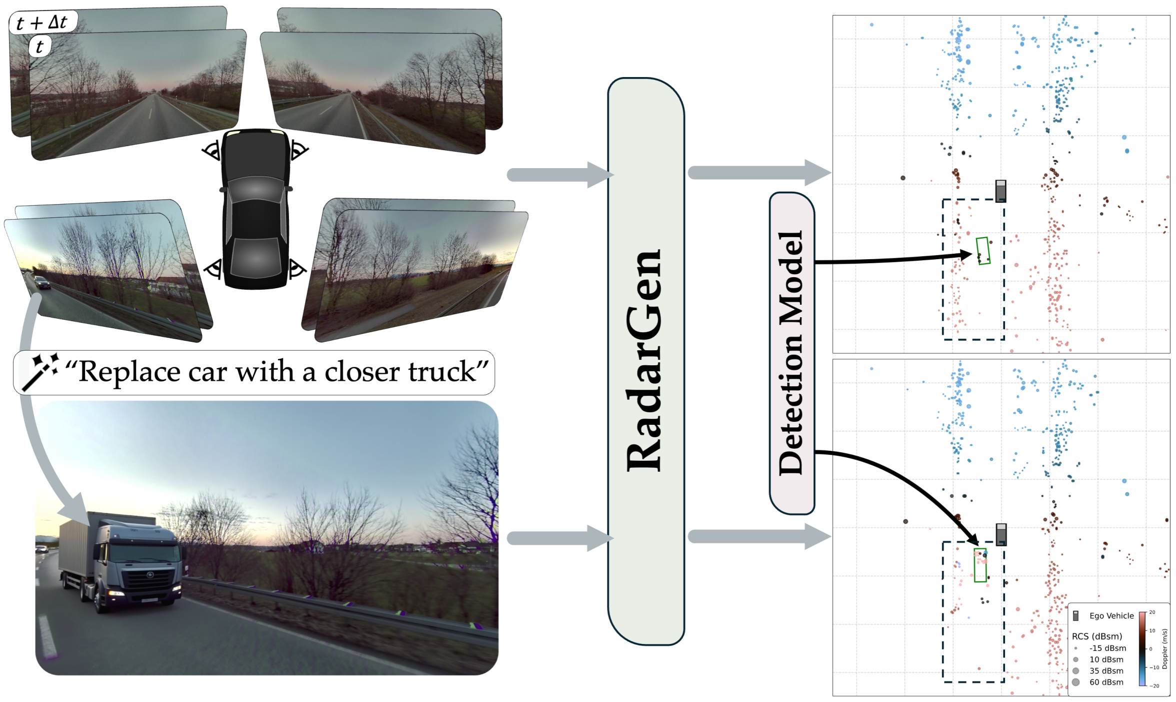

Figure 1. **Controllable radar synthesis from vision.** (Top) Given multi-view camera images, *RadarGen* generates realistic radar point clouds that align with real-world radar statistics and can be consumed by downstream perception models. (Bottom) The generation is semantically consistent: modifying the input scene with an off-the-shelf image editing tool (e.g., replacing a distant car with a closer truck) updates the radar response, removing returns from newly occluded regions and reflecting the new object geometry.

Abstract

We present RadarGen, a diffusion model for synthesizing realistic automotive radar point clouds from multi-view camera imagery. RadarGen adapts efficient image-latent diffusion to the radar domain by representing radar measurements in bird's-eye-view form that encodes spatial structure together with radar cross section (RCS) and Doppler attributes. A lightweight recovery step reconstructs point clouds from the generated maps. To better align generation with the visual scene, RadarGen incorporates BEV-aligned depth, semantic, and motion cues extracted from pretrained foundation models, which guide the stochastic generation process toward physically plausible radar patterns. Conditioning on images makes the approach broadly compatible, in principle, with existing visual datasets and simulation frameworks, offering a scalable direction for multi-modal generative simulation. Evaluations on large-scale driving data show that RadarGen captures characteristic radar measurement distributions and reduces the gap to perception models trained on real data, marking a step toward unified generative simulation across sensing modalities.

1. Introduction

Recent advances in neural and generative simulation have made it increasingly practical to synthesize photorealistic data at scale for autonomous driving. By reconstructing real scenes with neural fields or generating entirely new ones using video diffusion models, these systems can produce diverse and controllable environments that closely mimic real sensor observations. This capability enables large scale resimulation of traffic, lighting, and weather conditions without costly rerecording or manual setup [43, 51, 62]. Despite this rapid progress, most neural simulators remain limited to the visual domain, focusing on the generation of RGB imagery and video. Recent efforts have begun extending these ideas to LiDAR, demonstrating controllable three-dimensional point cloud generation from camera inputs [57, 60, 90]. Radar, however, remains an open frontier. Although it is already ubiquitous in production vehicles, providing low cost, lightweight, and weather resilient perception, it has received far less attention within the generative modeling community. This imbalance limits the fidelity of current neural simulators, which cannot reproduce radar's distinctive sensing characteristics, including signal sparsity, radar cross section (RCS), and Doppler.

Generating radar data poses unique challenges. Radar measurements exhibit strong stochasticity due to multipath reflections, interference, and material-dependent scattering that vary with scene geometry. Operating at longer wavelengths, radar interacts with surfaces and internal structures beyond what cameras or LiDAR perceive, making it highly complementary yet difficult to model from vision alone. A further challenge lies in the nature of available radar data. In most large scale driving datasets, radar is provided only after proprietary signal processing that converts raw radio frequency waveforms into sparse point clouds with RCS and Doppler values. This processing chain, which includes range Doppler transforms, beamforming, and detection algorithms such as constant false alarm rate (CFAR), is closed and lossy, discarding phase and other fine grained signal information. In practice, storing raw radar signals is extremely memory intensive, so even commercial survey vehicles often record only processed point clouds. As a result, point clouds remain the practical representation of radar data for large scale learning.

To address these challenges, we propose *RadarGen*, a generative framework for synthesizing automotive radar point clouds directly from camera imagery. *RadarGen* learns a distribution over radar observations conditioned on the visual scene, producing diverse and semantically consistent measurements rather than a single deterministic prediction, reflecting the inherent stochasticity of real radar signals. Conditioning on images allows *RadarGen* to leverage existing visual data and simulators, providing a scalable and modular way to enrich them with realistic radar signals.

A key design choice in *RadarGen* is to build on SANA [83], an efficient image-latent diffusion model proven effective and scalable for image synthesis. To the best of our knowledge, no existing generative model can reliably support scene level point cloud synthesis from multiview images of real driving scenes, making an image diffusion backbone a natural and practical foundation. SANA's architecture supports conditioning on visual input while efficiently handling the large number of tokens introduced by the multi-channel radar representation and by the additional conditioning cues incorporated later in our framework, allowing *RadarGen* to remain computationally efficient and deployable in large scale simulation settings.

To make radar data compatible with the image diffusion backbone, we express each radar point cloud as an image-like bird's eye view (BEV) representation. This representation encodes spatial structure together with radar cross section (RCS) and Doppler attributes, enabling all radar channels to share a unified latent space and allowing SANA's pretrained autoencoder to operate without modification. A lightweight smoothing and recovery step preserves geometric accuracy while ensuring stable latent encoding.

Learning radar purely from images is inherently difficult, as it requires reasoning about depth, semantics, and motion. To avoid forcing the denoiser to learn these cues from scratch, *RadarGen* leverages pretrained foundation models that provide dense visual priors. Depth, semantic, and motion cues extracted from each camera view are projected into bird's eye view to align with the radar representation and are fused within the diffusion model. We summarize our main contributions below.

- We present *RadarGen*, the first probabilistic diffusion framework to generate realistic automotive radar point clouds including location, RCS, and Doppler from multiview camera inputs.
- We introduce a latent diffusion methodology that trains on a BEV representation of sparse radar attributes, conditioned by BEV-aligned visual depth, semantic, and motion priors from foundation models.
- We establish comprehensive metrics for radar point clouds based on geometric fidelity, radar attribute fidelity, and distribution similarity, validating our design through extensive ablations.
- We demonstrate that the generated radar data can be interpreted by detectors trained on real data and supports applications such as scene editing.

2. Related Work

2.1. Physics-based radar simulation

Physics-based simulators model the emission, propagation and reception of electromagnetic (EM) waves based on physical laws. To rigorously simulate time-domain EM propagation, some works directly solve Maxwell's equations in the integral [10] or differential [25, 44, 71] form, offering high physical accuracy but are computation-intensive for real-time applications. For practical usage, other works approximate EM wave propagation based on geometric optics and ray-tracing [86]. This technique has been widely adapted to enhance system-level fidelity [22, 29, 33, 74], improve scalability and real-time performance [31, 68, 72], and support specific modern applications [3, 28, 67]. Commercial softwares also implement similar techniques [21, 59]. RadSimReal [5] utilized *standard* 3D reflection signals to lessen dependence on radar-specific internals in ray-tracing simulators. Distinct from ray-tracing, graphics-based simulators [52, 66, 70] exploit the rasterization pipeline of graphics engines as an efficient physics proxy, generating radar measurements from depth in real time. However, the aforementioned physical simulators rely on manually created assets, demanding substantial engineering effort to cover long-tail conditions.

2.2. Data-driven radar simulation

The heavy engineering efforts of physics-based radar simulators has motivated data-driven alternatives. One line adapts NeRF [6, 35, 41, 47, 56] or 3D Gaussian Splatting [37, 39, 42] to radar, learning scene-specific representations for novel-view rendering. These reconstructions, however, require multi-view radar captures per scene and transfer poorly to unseen scenes. In contrast, generative radar simulation methods synthesize measurements directly from conditioning inputs, enabling controllable generation for novel, unobserved environments. Prior works employ GANs [26] and VAEs [38] conditioned on object distance [24], scene layouts [16, 78], elevation maps [77], or LiDAR [42], but they ignore images as conditioning.

Two recent works [11, 17] use visual conditioning to radar generation, but focus on human-centric scenarios and depend on explicit physical modelling of signal-scene interactions, which limits scalability. For autonomous driving, Xiao et al. [82] synthesize radar cube with a U-Net conditioned on camera/LiDAR plus waveform-parameter embeddings, while Rangaraj et al. [58] generate range–azimuth maps using an autoencoder conditioned on depth and segmentation. Yet these methods target radar raw data that is unavailable for large-scale training [20]. Closer to our setting, Song et al. [69] and Alkanat et al. [2] produce radar point clouds from LiDAR/RGB with convolutional networks, but their deterministic mappings under-model radar stochasticity and also do not exploit large pretrained foundation models for comprehensive scene encoding.

2.3. Generative point cloud models

Generative models specific to radar point clouds remain scarce in literature, so we review general point cloud generative frameworks as references. Many works designed unconditional models [1, 9, 45, 48, 76, 85, 87, 89] for point cloud completion and generation or text-conditioned [50, 81] generative frameworks, which rely on large priors and lack spatial grounding and controllability. In contrast, some recent works [40, 46, 75] learn to generate point cloud conditioned on RGB images. However, these works are object-centric, only generating point cloud for object shapes. Instead, some methods are proposed to tackle scene-level generation of LiDAR point cloud with generative techniques like GAN [7, 63], VQ-VAE [84] and diffusion model [34, 49, 57, 80, 90]. While effective, these methods usually lack image-based conditional design and work on dense LiDAR range image, which cannot transfer to sparse and non-uniform sampled radar point clouds [18, 19, 53].

3. Preliminaries

We start by reviewing the challenges in generating radar point clouds (Sec. 3.1), and how we can benefit from efficient diffusion models (Sec. 3.2). This sets the stage for our proposed radar point cloud generation model (Sec. 4).

3.1. Challenges in radar point cloud generation

The key challenge in generative modeling of radar point clouds lies in their unique data characteristics. Radar point clouds are sparse, unordered 3D point sets with highly non-uniform sampling. Unlike LiDAR, whose returns are uniform along angular dimensions and can be reshaped into dense range images, radar detections arise from peak-based target extraction (e.g., CFAR [65]), and cannot form dense, grid-aligned measurements. Further, each radar point carries sensor-specific attributes beyond 3D position—Radar Cross-Section (RCS), a proxy for reflectivity, mainly affected by the object material, geometry and incident angles, and Doppler velocity, the measurement of relative radial velocity. As a result of this sparse, non-grid structure, we cannot effectively represent radar data in a range-image format. We thus adopt a multi-image bird's-eye-view (BEV) representation as a scalable alternative that is well-suited to the data's sparse nature.

3.2. Efficient diffusion models

A significant challenge in generative modeling is the synthesis of large-scale, explicit 3D point clouds conditioned on images. As discussed in Sec. 3.1 we use a multi-image BEV representation, which is inherently high-dimensional, posing a significant challenge for standard diffusion architectures. To tackle this high-dimensional generation problem at a scene-level scale, we must leverage an efficient diffusion architecture. Specifically, Latent Diffusion Models (LDMs) [32, 61] which compress images into smaller representations. Standard LDMs are insufficient, as they typically use an autoencoder (AE) with a 8x downsampling

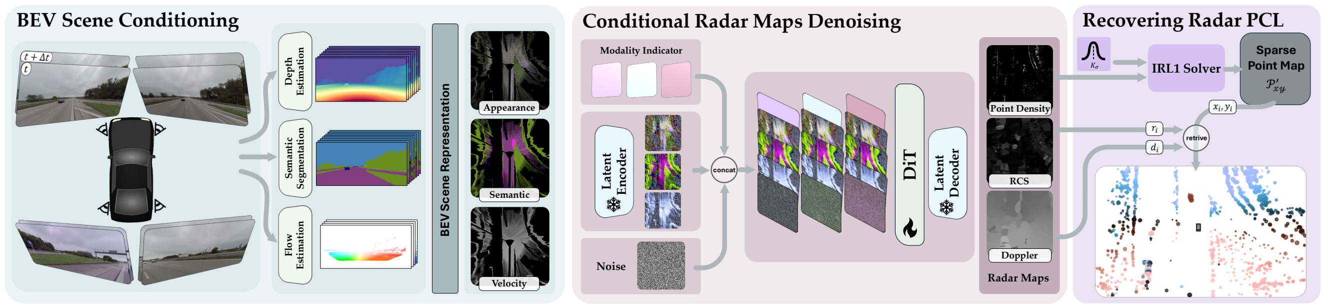

Figure 2. **Overview of RadarGen.** (Left) Multi-view posed images at time t and $t + \Delta t$ are fed through foundation model of metric depth estimation [55], semantic segmentation [13], and optical flow [88], enabling projection of the scene to BEV, encoding different information through color. (Middle) Encoded BEV representation is concatenated with a modality indicator specifying which map type to generate. During inference, the map is initialized as noise; during training, noise is added to the GT maps, and the Latent Encoder/Decoder are frozen while SANA's DiT [54, 83] is fine-tuned. (Right) During inference, the generated Point Density Map is deconvolved using an IRL1 Solver. The resulting sparse map is used to retrieve the RCS and Doppler values at corresponding locations to yield the final generated radar point cloud. Point color represents Doppler and point size represents RCS.

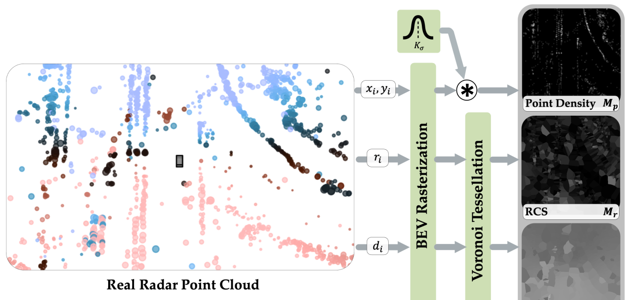

Figure 3. **Overview of representing radar as images (Sec. 4.1).** Constructing radar maps from a radar point cloud requires first rasterizing each point to BEV. The point locations are then convolved with a Gaussian kernel K_σ to produce the Point Density Map M_p. A Voronoi tessellation is also constructed, where each cell inherits the RCS and Doppler attributes from its corresponding point, producing the maps M_r and M_d respectively. Point color represents Doppler and point size represents RCS.

factor and a diffusion backbone with $O(N^2)$ quadratic self-attention complexity. We therefore build upon SANA [83], a framework that achieves efficiency by employing an AE with 32x compression and replacing the costly self-attention with an $O(N)$ linear attention mechanism.

4. Method

Problem statement. We address the task of surround-camera to radar point cloud generation. Given scene imagery captured by a rig of N outward-facing cameras $\mathbf{I}^t = \{I_1^t, \ldots, I_N^t\}$ at two consecutive timesteps $(t, t + \Delta t)$, together with known camera intrinsics and extrinsics, the goal is to generate a radar point cloud $\mathcal{P}^t = \{(x_i, y_i, r_i, d_i)\}_{i=1}^L$ representing the radar detections at time t in the ego-vehicle coordinate frame. Each element describes the planar spatial coordinates (x, y), radar cross section r (RCS), and Doppler velocity d.

Method overview. *RadarGen* consists of three main components (See Fig. 2 and 3). During training: (1) the conversion of the *sparse* radar point cloud $\mathcal{P}^t$ into a set of *dense* BEV representations suitable for a latent image diffusion model (Sec. 4.1); and (2) conditioning the generator on geometric and semantic cues derived from pretrained vision models to enhance structural and semantic understanding (Sec. 4.2). With these we learn the conditional distribution $p_\theta(\mathcal{P}^t \mid \mathbf{I}^t, \mathbf{I}^{t+\Delta t})$, which models the stochastic mapping from multi-view imagery to the radar point cloud at time t. At inference: (3) the recovery of the final sparse radar point cloud from the generated dense BEV predictions (Sec. 4.3).

4.1. Representing radar as images

Our goal is to train a diffusion denoiser that operates in the latent space of a pretrained autoencoder. To keep the autoencoder intact, the radar observations must therefore be represented as image-like signals that fit naturally within its latent distribution. However, radar measurements are sparse point clouds, far from the dense appearance statistics of natural images. We address this gap by transforming each radar point cloud into a set of dense, two-dimensional BEV maps that preserve radar-specific attributes while remaining compatible with the image-based AE. Fig. 3 illustrates the following steps.

Each radar point cloud is first projected onto the BEV plane by discarding elevation, which carries little discriminative information due to radar's limited vertical resolution in automotive setups. The projected points are then rasterized to form a BEV point map in which detections correspond to occupied pixels.

We construct three single-channel BEV maps: a Point

Density Map (M_p), an RCS Map (M_r), and a Doppler Map (M_d). The Point Density Map M_p is obtained by convolving the sparse BEV point map $\mathcal{P}_{xy}$ with a Gaussian kernel K_σ of variance σ, i.e., $M_p = K_\sigma * \mathcal{P}_{xy}$, producing a smooth estimate of radar return density. Increasing σ yields smoother maps with higher reconstruction fidelity under an AE, but also complicates point cloud recovery (see Sec. 4.3). For the RCS and Doppler maps, M_r and M_d, each pixel inherits the attribute value (RCS or Doppler) of the nearest radar detection, yielding a piecewise-constant map defined by the Voronoi tessellation of the detections.

To ensure compatibility with the autoencoder's RGB input space, each BEV map is replicated across three channels. We then encode each of these images independently, obtaining the latent representations z_p, z_r, and z_d that serve as the supervision targets for training the diffusion denoiser.

Additional preprocessing details, including region-of-interest cropping and grid resolution, are provided in Sec. 4.4.

4.2. Conditional radar generation

To simplify the conditional radar generation task, we employ pretrained vision foundation models to extract relevant factors from conditional inputs and project them into BEV representations which are semantically rich and spatially aligned with the radar BEV targets.

Specifically, we use a depth estimation network [55] to provide geometric cues, a semantic segmentation network [13] to supply categorical context, and an optical flow model [88] to infer per-pixel motion. These cues are projected and fused into a unified bird's-eye-view scene representation that serves as the conditioning input to the radar generator.

BEV scene conditioning. The conditioning representation, denoted $\mathbf{c}$, comprises three BEV maps: an Appearance map, a Semantic map, and a Radial Velocity map (see Fig. 2). We first project the N outward-facing camera images $\mathbf{I}^t = \{I_1^t, \dots, I_N^t\}$ into BEV using predicted metric depth [55]. For each image I_k^t, a depth estimation model provides a dense point map $P_{I_k^t}$, which is then transformed into the ego-vehicle coordinate frame using the known camera intrinsics and extrinsics. All point maps are merged and rasterized onto a common BEV grid to form the geometric backbone of the condition.

The points that construct the *Appearance* and *Semantic maps* obtain their color from the original images and segmented images, respectively. Using color-coded semantics, rather than one-hot encodings, ensures that the conditioning maps retain image-like statistics compatible with pretrained image encoders used later in our pipeline.

The *Radial Velocity map* provides motion cues analogous to Doppler velocity. To construct it, we first estimate optical flow between consecutive frames $(I^t, I^{t+\Delta t})$ using a pretrained flow network [88]. Let $\mathbf{f}(x)$ denote the flow vector at pixel x in I^t. Using the predicted depth at time t and $t + \Delta t$, we backproject x in I^t and its corresponding pixel $x + \mathbf{f}(x)$ in $I^{t+\Delta t}$ to 3D points $p^t(x)$ and $p^{t+\Delta t}(x)$ in the ego-vehicle frame. We then approximate the 3D velocity as $v(x) \approx \frac{p^{t+\Delta t}(x) - p^t(x)}{\Delta t}$, and retain only its component along the radial direction from the ego-vehicle to obtain a Doppler-like value. These radial velocities are rasterized and aggregated into a BEV grid, yielding the Radial Velocity conditioning map.

This BEV scene representation provides pixel-aligned conditioning for the radar diffusion model, ensuring that geometric, semantic, and dynamic cues are spatially consistent with the generated radar BEV maps.

Conditional radar maps denoising. With the radar maps represented in latent space, the learning objective becomes to model the distribution of plausible radar latents conditioned on the visual scene. We therefore train a conditional diffusion denoiser that learns $p(z_p, z_r, z_d \mid \mathbf{c})$, where $\mathbf{c}$ denotes the BEV conditioning maps. Because both the conditioning and target latents are defined in a spatially aligned BEV coordinate frame, the conditioning tensors can be concatenated directly along the channel dimension [36], providing the model with a compact yet expressive geometric prior. This formulation allows the network to generate diverse yet physically consistent radar realizations aligned with the observed camera views.

The denoiser ϵ_θ is implemented as a Diffusion Transformer (DiT) [54] that operates jointly on the latent representations of the three radar maps—density, RCS, and Doppler. During each denoising step, the latents for these maps are concatenated into a single token sequence and processed through shared self-attention, enabling the network to capture correlations across radar modalities. Modality-specific identifiers, defined as learnable embeddings m_i [30] concatenated feature-wise to each radar map $i \in \{p, r, d\}$, guide the transformer to preserve the distinct statistics of each channel while still sharing information between them. At inference, Gaussian noise is iteratively denoised under BEV conditioning to produce the latent radar maps $\{z'_p, z'_r, z'_d\}$, which are then decoded into the final BEV radar images by the pretrained decoder $\mathcal{D}$.

4.3. Recovering the sparse radar point cloud

As described in Sec. 4.1, the point density map is modeled as the convolution of a sparse point map with a Gaussian kernel, $M_p = K_\sigma * \mathcal{P}_{xy}$. The generative model therefore produces a blurred density estimate M'_p, from which the underlying discrete point locations must be recovered.

We formulate recovery as an explicit deconvolution problem, taking advantage of the fact that the blurring kernel K_σ is known and fixed, since it was used to generate the training targets. This knowledge of the forward blurring

Table 1. **Quantitavie evaluation.** *RadarGen* broadly outperforms the baseline on geometric fidelity (CD, IoU, Density Similarity, Hit Rate), radar attribute fidelity (DA Recall, Precision, F1), and distribution similarity (MMD).

Method	**Entire Area**								
	CD Loc. (↓)	**CD Full (↓)**	**IoU@1m (↑)**	**DA Recall (↑)**	**DA Prec. (↑)**	**DA F1 (↑)**	**MMD Loc. (↓)**	**MMD RCS (↓)**	**MMD Doppler (↓)**
Baseline	1.84 ± 0.48	$\mathbf{0.038 \pm 0.009}$	0.23 ± 0.10	0.15 ± 0.10	0.14 ± 0.10	0.14 ± 0.09	0.368 ± 0.151	0.36 ± 0.25	0.65 ± 0.64
RadarGen	$\mathbf{1.68 \pm 0.39}$	0.040 ± 0.008	$\mathbf{0.31 \pm 0.11}$	$\mathbf{0.23 \pm 0.12}$	$\mathbf{0.26 \pm 0.12}$	$\mathbf{0.24 \pm 0.12}$	$\mathbf{0.056 \pm 0.062}$	$\mathbf{0.09 \pm 0.15}$	$\mathbf{0.31 \pm 0.74}$

	Foreground												
	CD Loc. (↓)	**CD Full (↓)**	**Density Sim. (↑)**	**Hit Rate (↑)**	**MMD Car (↓)**			**MMD Truck (↓)**			**MMD Trailer (↓)**		
					Loc.	**RCS**	**Doppler**	**Loc.**	**RCS**	**Doppler**	**Loc.**	**RCS**	**Doppler**
Baseline	1.32 ± 0.79	0.075 ± 0.049	0.35 ± 0.43	0.37	**0.035**	0.753	0.549	0.167	0.202	0.485	0.0459	0.064	0.607
RadarGen	$\mathbf{0.95 \pm 0.65}$	$\mathbf{0.069 \pm 0.049}$	$\mathbf{0.51 \pm 0.41}$	**0.66**	0.037	**0.006**	**0.014**	**0.024**	**0.031**	**0.060**	**0.0069**	**0.022**	**0.046**

process enables us to pose a well-defined inverse problem for recovery, yielding a principled and controllable reconstruction of the sparse radar detections.

We solve for the sparse point map $\mathcal{P}'_{xy}$ via an L1-regularized, non-negative deconvolution (LASSO) [73]:

$$\min_{\mathcal{P}_{xy}\geq 0} \frac{1}{2}\|K_\sigma * \mathcal{P}_{xy} - M'_p\|_2^2 + \lambda\|\mathcal{P}_{xy}\|_1, \tag{1}$$

where λ balances data fidelity and sparsity. We optimize this objective using an Iteratively Reweighted L1 (IRL1) [15] scheme with a FISTA [4] solver, which provides fast convergence and stable recovery of sparse signals. The resulting $\mathcal{P}'_{xy}$ is thresholded to extract the final set of L 2D coordinates $\{(x_i, y_i)\}_{i=1}^{L}$. For each recovered location, we retrieve the RCS and Doppler attributes from their corresponding map positions, producing the final reconstructed radar point cloud $\mathcal{P}' = \{(x_i, y_i, r_i, d_i)\}_{i=1}^{L}$.

4.4. Implementation details

Our diffusion model is a modified SANA DiT [54, 83] adapted for image conditioning, utilizing its pre-trained v1.1 autoencoder. We use UnidepthV2 [55] for metric depth estimation, Mask2Former [13] (trained on Cityscapes [14]) for semantic segmentation, and UniFlow [88] for flow prediction. We trained the model for 2 days on 8 L40 (48GB) GPUs. During training, we drop each condition with a 10% probability. We filter the radar point clouds to a $\pm$50m range and use a 512×512 grid. Further implementation details are in Sec. B.

5. Experiments

We start by describing the the dataset, metrics, and baseline. In Sec. 5.1, we evaluate *RadarGen* on radar point cloud generation from images. We then show how our proposed method can be used for scene editing in Sec. 5.2. Finally, in Sec. 5.3 we analyze our design choices.

Dataset. We evaluate on the MAN TruckScenes dataset [23] of driving scenes, which provides multi-view camera images and radar data. After filtering out low-light night scenes, we use 431 clips for training and 49 for evaluation. While clips contain approximately 200 frames each, only 40 include bounding box annotations. We train on all frames in the training set and evaluate only on the annotated frames, which include bounding boxes for visible objects.

Evaluation metrics. As no standard benchmarks exist for our conditional radar generation task, we propose an evaluation framework. Our metrics assess three key aspects: geometric fidelity, radar attribute fidelity, and distribution similarity. Given the importance of foreground points for object detection, we calculate metrics for both the overall scene and points within annotated object bounding boxes. For geometric fidelity, we report *Chamfer Distance* on location (CD Loc.) and the full vector of normalized location, RCS and Doppler (CD Full), Point Cloud *IoU* at 1m (IoU@1m) [64], *Density Similarity*, which measures the distance in number of points in a bounding box compared to the ground truth, and Bounding Box *Hit Rate*. Radar attribute fidelity is assessed using *Distance-Attribute* (DA) Recall, Precision, and F1-score, which assess the proximity of generated points with similar RCS and Doppler values to ground truth points. Finally, we evaluate distribution similarity using *Maximum Mean Discrepancy* (MMD) for both entire point clouds and for points aggregated within object bounding boxes across scenes. Additional details on all metrics are available in Sec. A.

Baseline. We use the feedforward model RGB2Point [40], which is suitable for the task of prediction a point cloud from multi-view images. We extend the model output to include RCS and Doppler, and increase the number of parameters to 432M, which is comparable to our model with 592M parameters. See Sec. B.4 for more implementation details about this baseline.

5.1. Radar point cloud generation

Quantitative evaluation. In Tab. 1 we compare *RadarGen* to the baseline on the MAN TruckScenes dataset. *RadarGen* broadly outperforms the baseline. A notable exception is CD Full in Entire Area, which is expected as the baseline model was trained using a similar loss objective.

Qualitative evaluation. In Fig. 4, we visually compare our results with the baseline and ground truth on two ex-

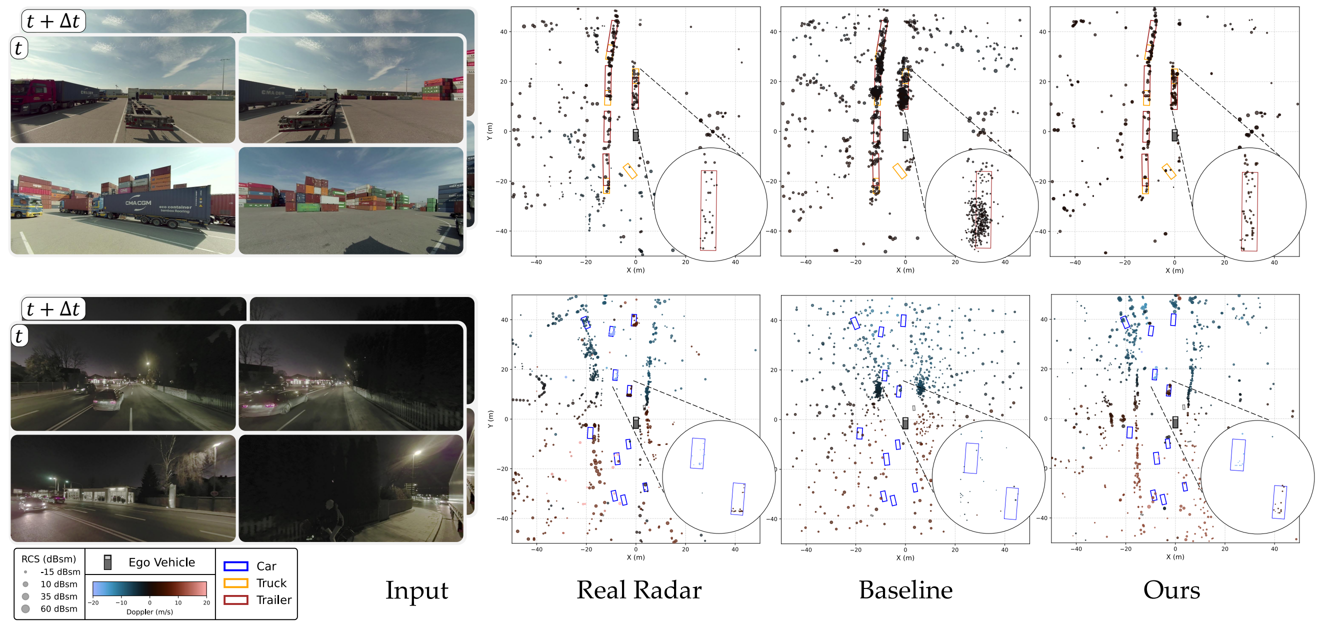

Figure 4. **Qualitative results.** Our model generates point clouds with higher geometric and attribute fidelity to the ground truth compared to the baseline. *RadarGen* uses inputs t and $t + \Delta t$, while the baseline uses only t. Ground truth bounding boxes are highlighted in color.

amples. *RadarGen's* generated point clouds closely match the ground truth in shape, distribution, and count, demonstrating a significant advantage over the baseline. Further visualizations are available in Sec. C.

Compatibility with perception models. We adapt and train the VoxelNeXt [12] detector on MAN TruckScenes radar data, which yields an NDS [8] of 0.48 on real data (50m range). We then evaluated this detector on generated radar point clouds (PCL). On PCLs from *RadarGen*, the detector scored NDS of 0.30. In contrast, the detector struggled to find valid objects in the baseline's PCLs, resulting in NDS of nearly zero. Full results are available in Sec. C.4. Interestingly, while our generated radar achieves a high hit rate (0.66) within true bounding boxes, the overall detection quality still underperforms compared to real data. This could be attributed to the detector's tailoring to specific, intricate properties of the true data that our model may not fully capture. A detailed analysis of the subtle differences between real and generated radar data is beyond the scope of this paper.

Figure 1 visualizes detections on our generated PCL and demonstrates an augmentation scenario. In this example, a real vehicle is replaced with a generated truck; the detector successfully perceives the newly generated points as a truck.

5.2. Scene editing

Our method supports radar point cloud augmentation by editing the input images using an off-the-shelf image editing tools, such as ChronoEdit [79]. Fig. 1 demonstrates

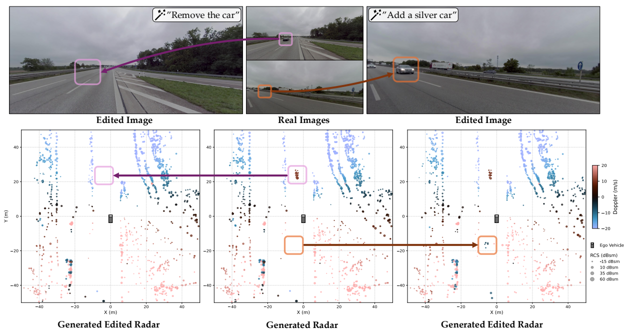

Figure 5. **Scene editing.** Modifying the input images using an off-the-shelf image editing tool updates the radar response, demonstrating object removal (left) and insertion (right).

object replacement, while Fig. 5 shows examples of object removal and insertion. Notably, in the Fig. 1 replacement example (car to truck), the model correctly removes radar points from the area newly occluded by the truck, demonstrating that it properly handles occlusion changes.

5.3. Ablation study and analysis

BEV conditioning ablations We assess the contribution of each BEV condition, by zeroing it out. Results are shown in Tab. 2. Removing the segmentation map causes the most significant degradation. It worsens geometric fidelity and significantly increases the RCS MMD for both the Entire Area and per-object class in the Foreground. Ablating either the velocity map or the appearance map primarily degrades the Doppler MMD. Notably, the degradation from remov-

Table 2. **Ablation Study.** We demonstrate the importance of each *RadarGen* condition and compare against a model conditioned directly on multi-view (MV) camera images. Evaluation covers geometric fidelity (CD, IoU, Density Similarity, Hit Rate), radar attribute fidelity (DA Recall, Precision, F1), and distribution similarity (MMD).

Method	Entire Area								
	CD Loc. (↓)	**CD Full (↓)**	**IoU @ 1m (↑)**	**DA Recall (↑)**	**DA Prec. (↑)**	**DA F1 (↑)**	**MMD Loc. (↓)**	**MMD RCS (↓)**	**MMD Doppler (↓)**
MV Camera Cond.	1.88 ± 0.50	0.041 ± 0.009	0.28 ± 0.12	0.26 ± 0.13	0.25 ± 0.12	0.25 ± 0.11	0.059 ± 0.057	0.06 ± 0.09	0.24 ± 0.85
W/o Appearance map	1.71 ± 0.40	0.040 ± 0.008	0.31 ± 0.11	0.23 ± 0.12	0.25 ± 0.12	0.23 ± 0.12	0.059 ± 0.069	0.09 ± 0.16	0.35 ± 0.86
W/o Semantic map	1.72 ± 0.40	0.041 ± 0.010	0.31 ± 0.11	0.22 ± 0.12	0.24 ± 0.12	0.23 ± 0.12	0.059 ± 0.061	0.12 ± 0.26	0.33 ± 0.72
W/o Velocity map	1.69 ± 0.40	0.040 ± 0.008	0.31 ± 0.11	0.23 ± 0.12	0.25 ± 0.12	0.23 ± 0.12	0.057 ± 0.064	0.09 ± 0.16	0.34 ± 0.80
RadarGen	1.68 ± 0.39	0.040 ± 0.008	0.31 ± 0.11	0.23 ± 0.12	0.26 ± 0.12	0.24 ± 0.12	0.056 ± 0.062	0.09 ± 0.15	0.31 ± 0.74

	Foreground												
	CD Loc. (↓)	**CD Full (↓)**	**Density Sim. (↑)**	**Hit Rate (↑)**	**MMD Car (↓)**			**MMD Truck (↓)**			**MMD Trailer (↓)**		
					Loc.	**RCS**	**Doppler**	**Loc.**	**RCS**	**Doppler**	**Loc.**	**RCS**	**Doppler**
MV Camera Cond.	1.0 ± 0.67	0.069 ± 0.050	0.47 ± 0.42	0.56	0.025	0.018	0.024	0.017	0.070	0.073	0.0029	0.037	0.040
W/o Appearance map	0.95 ± 0.64	0.069 ± 0.050	0.51 ± 0.41	0.65	0.034	0.006	0.019	0.019	0.019	0.059	0.0048	0.026	0.041
W/o Semantic map	0.96 ± 0.66	0.070 ± 0.050	0.50 ± 0.41	0.64	0.045	0.010	0.017	0.023	0.039	0.078	0.0072	0.032	0.061
W/o Velocity map	0.95 ± 0.65	0.069 ± 0.049	0.51 ± 0.41	0.66	0.033	0.006	0.019	0.024	0.018	0.070	0.0051	0.024	0.047
RadarGen	0.95 ± 0.65	0.069 ± 0.049	0.51 ± 0.41	0.66	0.037	0.006	0.014	0.024	0.031	0.060	0.0069	0.022	0.046

ing the appearance map, even with the segmentation map present, suggests the model leverages finer-grained appearance details to refine its understanding of object class and thus generate a more realistic motion profile. This confirms that all input channels are meaningfully fused to produce radar point clouds.

Comparison to multi-view conditioning. We also compare against a model conditioned directly on multi-view (MV) camera images, which omits the BEV input. Instead, it uses images from t and $t + \Delta t$ concatenated as input tokens with plucker and modality embeddings. While this MV model achieves an improved MMD for radar attributes on the Entire Area, our BEV conditioned model yields better overall geometric fidelity, as detailed in Tab. 2. Furthermore, the MV approach is computationally prohibitive, requiring over $3\times$ runtime; this model trained for 9 days, compared to only 2 days for our BEV conditioned model. While our BEV representation provides a more efficient and geometrically accurate solution, the results from the MV model highlight it as a valuable direction for future research.

Radar PCL reconstruction. We ablate two factors: (a) the 2D Gaussian kernel bandwidth σ for the Point Density Map, and (b) the PCL recovery method. We test $\sigma \in \{0.5, 1, 1.5, 2, 2.5, 3\}$ and four recovery methods (`random`, `peak`, `peak+random`, `deconv`) on the MANTruckScenes *mini-train* set, evaluating AE reconstruction error and the geometric fidelity of the recovered PCL. As shown in Fig. 6 (left), larger σ values reduce AE reconstruction error, as smoother maps are easier to reconstruct. However, overly large σ over-smooths the map structure, degrading downstream PCL recovery. Balancing this trade-off, we set $\sigma = 2$. For the recovery method, `deconv` consistently yields the best PCL quality across all σ values and for both ground-truth and AE-reconstructed maps, validating its ability to preserve the spatial distribution.

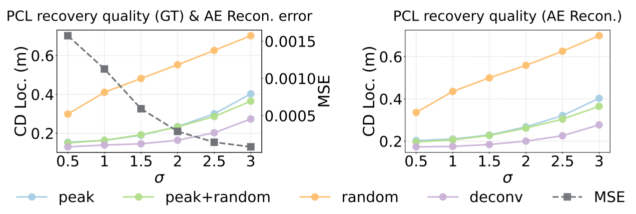

Figure 6. Impact of σ and recovery method on PCL reconstruction. `peak` selects local maxima; `random` samples pixels proportional to the probability map (without replacement); `peak+random` takes peaks first, then fills remaining points by probability sampling excluding the peaks. MSE is calculated between the input and output to AE, whose range is [-1,1].

6. Conclusion and Future Work

In this paper, we introduced *RadarGen*, a probabilistic diffusion framework for generating realistic automotive radar point clouds from multi-view camera inputs. Our method leverages foundation models to create a unified BEV representation for conditioning, generates dense BEV radar maps, and uses a deconvolution solver to recover the final sparse point cloud. Experimentally, *RadarGen* outperforms our proposed baseline on a comprehensive suite of geometric, attribute, and distribution metrics. We demonstrate its application in data augmentation via simple image editing and show promising results for downstream detectors. Future work will focus on extending our framework for video input, exploring text-based conditioning, and training on multiple datasets and radar configurations.

Limitations. Our method's performance is inherently tied to the capabilities of the upstream foundation models. It is thus limited in challenging scenarios where these models underperform, such as in low-light night scenes, with strong reflections, or during camera occlusion. Additionally, our model can generate points in areas not directly visible to the cameras. This behavior is a double-edged sword: while it is desirable for "filling in" occluded objects, it can also lead to uncontrolled hallucinations in these unseen regions.

Acknowledgments

Or Litany acknowledges support from the Israel Science Foundation (grant 624/25) and the Azrieli Foundation Early Career Faculty Fellowship. This research was also supported in part by an academic gift from Meta. The authors gratefully acknowledge this support.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *ICML*, 2018. 3

[2] Tunc Alkanat and Ashish Pandharipande. Automotive radar point cloud parametric density estimation using camera images. In *ICASSP*, 2024. 3

[3] Maximilian Arnold, Maximilian Bauhofer, Silvio Mandelli, Marcus Henninger, Frank Schaich, Thorsten Wild, and Stephan ten Brink. Maxray: A raytracing-based integrated sensing and communication framework. In *IEEE JC&S*, 2022. 3

[4] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2009. 6

[5] Oded Bialer and Yuval Haitman. Radsimreal: Bridging the gap between synthetic and real data in radar object detection with simulation. In *CVPR*, 2024. 3

[6] David Borts, Erich Liang, Tim Broedermann, Andrea Ramazzina, Stefanie Walz, Edoardo Palladin, Jipeng Sun, David Brueggemann, Christos Sakaridis, Luc Van Gool, et al. Radar fields: Frequency-space neural scene representations for fmcw radar. In *ACM SIGGRAPH*, 2024. 3

[7] Lucas Caccia, Herke Van Hoof, Aaron Courville, and Joelle Pineau. Deep generative modeling of lidar data. In *IROS*, 2019. 3

[8] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020. 7

[9] Ruojin Cai, Guandao Yang, Hadar Averbuch-Elor, Zekun Hao, Serge Belongie, Noah Snavely, and Bharath Hariharan. Learning gradient fields for shape generation. In *ECCV*, 2020. 3

[10] Carlo Capsoni and Michele D'Amico. A physically based radar simulator. *Journal of Atmospheric and Oceanic Technology*, 1998. 3

[11] Xingyu Chen and Xinyu Zhang. Rf genesis: Zero-shot generalization of mmwave sensing through simulation-based data synthesis and generative diffusion models. In *Proceedings of the 21st ACM Conference on Embedded Networked Sensor Systems*, 2023. 3

[12] Yukang Chen, Jianhui Liu, Xiangyu Zhang, Xiaojuan Qi, and Jiaya Jia. Voxelnext: Fully sparse voxelnet for 3d object detection and tracking. In *CVPR*, 2023. 7, 15

[13] Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *CVPR*, 2022. 4, 5, 6

[14] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 6

[15] Ingrid Daubechies, Ronald DeVore, Massimo Fornasier, and C Sinan Güntürk. Iteratively reweighted least squares minimization for sparse recovery. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 2010. 6

[16] Marcio L Lima De Oliveira and Marco JG Bekooij. Generating synthetic short-range fmcw range-doppler maps using generative adversarial networks and deep convolutional autoencoders. In *IEEE Radar Conference*, 2020. 3

[17] Kaikai Deng, Dong Zhao, Qiaoyue Han, Zihan Zhang, Shuyue Wang, Anfu Zhou, and Huadong Ma. Midas: Generating mmwave radar data from videos for training pervasive and privacy-preserving human sensing tasks. *Proceedings of the ACM on IMWUT*, 2023. 3

[18] Fangqiang Ding, Zhijun Pan, Yimin Deng, Jianning Deng, and Chris Xiaoxuan Lu. Self-supervised scene flow estimation with 4-d automotive radar. *IEEE RA-L*, 2022. 3

[19] Fangqiang Ding, Andras Palffy, Dariu M Gavrila, and Chris Xiaoxuan Lu. Hidden gems: 4d radar scene flow learning using cross-modal supervision. In *CVPR*, 2023. 3

[20] Fangqiang Ding, Xiangyu Wen, Yunzhou Zhu, Yiming Li, and Chris Xiaoxuan Lu. Radarocc: Robust 3d occupancy prediction with 4d imaging radar. *NeurIPS*, 2024. 3

[21] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, 2017. 3

[22] Manuel Dudek, Dietmar Kissinger, Robert Weigel, and Georg Fischer. A millimeter-wave fmcw radar system simulator for automotive applications including nonlinear component models. In *EuRAD*, 2011. 3

[23] Felix Fent, Fabian Kuttenreich, Florian Ruch, Farija Rizwin, Stefan Juergens, Lorenz Lechermann, Christian Nissler, Andrea Perl, Ulrich Voll, Min Yan, et al. Man truckscenes: A multimodal dataset for autonomous trucking in diverse conditions. *NeurIPS*, 2024. 6

[24] Eduardo C Fidelis, Fabio Reway, Herick Ribeiro, Pietro L Campos, Werner Huber, Christian Icking, Lester A Faria, and Torsten Schön. Generation of realistic synthetic raw radar data for automated driving applications using generative adversarial networks. *arXiv preprint arXiv:2308.02632*, 2023. 3

[25] Iraklis Giannakis, Antonios Giannopoulos, and Craig Warren. A realistic fdtd numerical modeling framework of ground penetrating radar for landmine detection. *IEEE journal of selected topics in applied earth observations and remote sensing*, 2015. 3

[26] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 2020. 3

[27] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *JMLR*, 2012. 14

[28] Demetrio Gubelli, Oleg A Krasnov, and Olexander Yarovyi. Ray-tracing simulator for radar signals propagation in radar networks. In *EuRAD*, 2013. 3

[29] Danping He, Ke Guan, Bo Ai, Zhangdui Zhong, Junhyeong Kim, Heesang Chung, and Andrej Hrovat. Channel measurement and ray-tracing simulation for 77 ghz automotive radar. *IEEE TITS*, 2022. 3

[30] Kai He, Ruofan Liang, Jacob Munkberg, Jon Hasselgren, Nandita Vijaykumar, Alexander Keller, Sanja Fidler, Igor Gilitschenski, Zan Gojcic, and Zian Wang. Unirelight: Learning joint decomposition and synthesis for video relighting. *arXiv preprint arXiv:2506.15673*, 2025. 5

[31] Nils Hirsenkorn, Paul Subkowski, Timo Hanke, Alexander Schaermann, Andreas Rauch, Ralph Rasshofer, and Erwin Biebl. A ray launching approach for modeling an fmcw radar system. In *IRS*, 2017. 3

[32] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arxiv:2006.11239*, 2020. 3

[33] Martin Holder, Clemens Linnhoff, Philipp Rosenberger, and Hermann Winner. The fourier tracing approach for modeling automotive radar sensors. In *IRS*, 2019. 3

[34] Qianjiang Hu, Zhimin Zhang, and Wei Hu. Rangeldm: Fast realistic lidar point cloud generation. In *ECCV*, 2024. 3

[35] Tianshu Huang, John Miller, Akarsh Prabhakara, Tao Jin, Tarana Laroia, Zico Kolter, and Anthony Rowe. Dart: Implicit doppler tomography for radar novel view synthesis. In *CVPR*, 2024. 3

[36] Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *CVPR*, 2024. 5

[37] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 2023. 3

[38] Diederik P Kingma, Max Welling, et al. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 2019. 3

[39] Pou-Chun Kung, Skanda Harisha, Ram Vasudevan, Aline Eid, and Katherine A Skinner. Radarsplat: Radar gaussian splatting for high-fidelity data synthesis and 3d reconstruction of autonomous driving scenes. *arXiv preprint arXiv:2506.01379*, 2025. 3

[40] Jae Joong Lee and Bedrich Benes. Rgb2point: 3d point cloud generation from single rgb images. In *WACV*, 2025. 3, 6, 15

[41] Zhengxin Lei, Feng Xu, Jiangtao Wei, Feng Cai, Feng Wang, and Ya-Qiu Jin. Sar-nerf: Neural radiance fields for synthetic aperture radar multi-view representation. *IEEE TGRS*, 2024. 3

[42] Aobo Li, Zhengxin Lei, Jiangtao Wei, and Feng Xu. Sargs: 3d gaussian splatting for synthetic aperture radar target reconstruction. *arXiv preprint arXiv:2506.21633*, 2025. 3

[43] Chih-Hao Lin, Zian Wang, Ruofan Liang, Yuxuan Zhang, Sanja Fidler, Shenlong Wang, and Zan Gojcic. Controllable weather synthesis and removal with video diffusion models. *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. 2

[44] Hai Liu, Bangan Xing, Honghua Wang, Jie Cui, and Billie F Spencer. Simulation of ground penetrating radar on dispersive media by a finite element time domain algorithm. *Journal of applied geophysics*, 2019. 3

[45] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *CVPR*, 2021. 3

[46] Luke Melas-Kyriazi, Christian Rupprecht, and Andrea Vedaldi. Pc2: Projection-conditioned point cloud diffusion for single-image 3d reconstruction. In *CVPR*, 2023. 3

[47] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 2021. 3

[48] Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy J Mitra, and Leonidas J Guibas. Structurenet: hierarchical graph networks for 3d shape generation. *ACM TOG*, 2019. 3

[49] Kazuto Nakashima and Ryo Kurazume. Lidar data synthesis with denoising diffusion probabilistic models. In *ICRA*, 2024. 3

[50] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*, 2022. 3

[51] NVIDIA, Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, Daniel Dworakowski, Jiaojiao Fan, Michele Fenzi, Francesco Ferroni, Sanja Fidler, Dieter Fox, Songwei Ge, Yunhao Ge, Jinwei Gu, Siddharth Gururani, Ethan He, Jiahui Huang, Jacob Huffman, Pooya Jannaty, Jingyi Jin, Seung Wook Kim, Gergely Klár, Grace Lam, Shiyi Lan, Laura Leal-Taixe, Anqi Li, Zhaoshuo Li, Chen-Hsuan Lin, Tsung-Yi Lin, Huan Ling, Ming-Yu Liu, Xian Liu, Alice Luo, Qianli Ma, Hanzi Mao, Kaichun Mo, Arsalan Mousavian, Seungjun Nah, Sriharsha Niverty, David Page, Despoina Paschalidou, Zeeshan Patel, Lindsey Pavao, Morteza Ramezanali, Fitsum Reda, Xiaowei Ren, Vasanth Rao Naik Sabavat, Ed Schmerling, Stella Shi, Bartosz Stefaniak, Shitao Tang, Lyne Tchapmi, Przemek Tredak, Wei-Cheng Tseng, Jibin Varghese, Hao Wang, Haoxiang Wang, Heng Wang, Ting-Chun Wang, Fangyin Wei, Xinyue Wei, Jay Zhangjie Wu, Jiashu Xu, Wei Yang, Lin Yen-Chen, Xiaohui Zeng, Yu Zeng, Jing Zhang, Qinsheng Zhang, Yuxuan Zhang, Qingqing Zhao, and Artur Zolkowski. Cosmos world foundation model platform for physical ai, 2025. 2

[52] Monika Ouza, Michael Ulrich, and Bin Yang. A simple radar simulation tool for 3d objects based on blender. In *IRS*, 2017. 3

[53] Zhijun Pan, Fangqiang Ding, Hantao Zhong, and Chris Xiaoxuan Lu. Ratrack: moving object detection and tracking with 4d radar point cloud. In *ICRA*, 2024. 3

[54] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023. 4, 5, 6

[55] Luigi Piccinelli, Christos Sakaridis, Yung-Hsu Yang, Mattia Segu, Siyuan Li, Wim Abbeloos, and Luc Van Gool. UniDepthV2: Universal monocular metric depth estimation made simpler, 2025. 4, 5, 6

[56] Mahan Rafidashti, Ji Lan, Maryam Fatemi, Junsheng Fu, Lars Hammarstrand, and Lennart Svensson. Neuradar: Neural radiance fields for automotive radar point clouds. In *CVPR*, 2025. 3

[57] Haoxi Ran, Vitor Guizilini, and Yue Wang. Towards realistic scene generation with lidar diffusion models. In *CVPR*, 2024. 2, 3

[58] Pavan Aakash Rangaraj, Tunc Alkanat, and Ashish Pandharipande. Raids: Radar range-azimuth map estimation from image, depth, and semantic descriptions. *IEEE Sensors Journal*, 2025. 3

[59] Remcom Inc. Wavefarer® automotive radar software, 2024. 3

[60] Xuanchi Ren, Yifan Lu, Tianshi Cao, Ruiyuan Gao, Shengyu Huang, Amirmojtaba Sabour, Tianchang Shen, Tobias Pfaff, Jay Zhangjie Wu, Runjian Chen, et al. Cosmos-drive-dreams: Scalable synthetic driving data generation with world foundation models. *arXiv preprint arXiv:2506.09042*, 2025. 2

[61] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021. 3

[62] Lloyd Russell, Anthony Hu, Lorenzo Bertoni, George Fedoseev, Jamie Shotton, Elahe Arani, and Gianluca Corrado. Gaia-2: A controllable multi-view generative world model for autonomous driving. *arXiv preprint arXiv:2503.20523*, 2025. 2

[63] Ahmad El Sallab, Ibrahim Sobh, Mohamed Zahran, and Nader Essam. Lidar sensor modeling and data augmentation with gans for autonomous driving. *arXiv preprint arXiv:1905.07290*, 2019. 3

[64] Kieran Saunders, Luis J Manso, and George Vogiatzis. Baseboostdepth: Exploiting larger baselines for self-supervised monocular depth estimation. *arXiv preprint arXiv:2407.20437*, 2024. 6, 13

[65] Louis L Scharf and Cédric Demeure. *Statistical signal processing: detection, estimation, and time series analysis*. Prentice Hall, 1991. 3

[66] Christian Schöffmann, Barnaba Ubezio, Christoph Böhm, Stephan Mühlbacher-Karrer, and Hubert Zangl. Virtual radar: Real-time millimeter-wave radar sensor simulation for perception-driven robotics. *IEEE RA-L*, 2021. 3

[67] Christian Schüßler, Marcel Hoffmann, Johanna Bräunig, Ingrid Ullmann, Randolf Ebelt, and Martin Vossiek. A realistic radar ray tracing simulator for large mimo-arrays in automotive environments. *IEEE Journal of Microwaves*, 2021. 3

[68] Arien P Sligar. Machine learning-based radar perception for autonomous vehicles using full physics simulation. *IEEE Access*, 2020. 3

[69] Peili Song, Dezhen Song, Yifan Yang, Enfan Lan, and Jingtai Liu. Simulating automotive radar with lidar and camera inputs. *arXiv preprint arXiv:2503.08068*, 2025. 3

[70] Christian Stetco, Barnaba Ubezio, Stephan Mühlbacher-Karrer, and Hubert Zangl. Radar sensors in collaborative robotics: Fast simulation and experimental validation. In *IEEE ICRA*, 2020. 3

[71] Mark L Stowell, Benjamin J Fasenfest, and Daniel A White. Investigation of radar propagation in buildings: A 10-billion element cartesian-mesh fetd simulation. *IEEE transactions on antennas and propagation*, 2008. 3

[72] Jörn Thieling, Susanne Frese, and Jürgen Roßmann. Scalable and physical radar sensor simulation for interacting digital twins. *IEEE Sensors Journal*, 2020. 3

[73] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 1996. 6

[74] Ali Eray Topak, Jürgen Hasch, and Thomas Zwick. A system simulation of a 77 ghz phased array radar sensor. In *IRS*, 2011. 3

[75] Michał J Tyszkiewicz, Pascal Fua, and Eduard Trulls. Gecco: Geometrically-conditioned point diffusion models. In *ICCV*, 2023. 3

[76] Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, Karsten Kreis, et al. Lion: Latent point diffusion models for 3d shape generation. *NeurIPS*, 2022. 3

[77] Rob Weston, Oiwi Parker Jones, and Ingmar Posner. There and back again: Learning to simulate radar data for real-world applications. In *IEEE ICRA*, 2021. 3

[78] Tim A Wheeler, Martin Holder, Hermann Winner, and Mykel J Kochenderfer. Deep stochastic radar models. In *IEEE IV*, 2017. 3

[79] Jay Zhangjie Wu, Xuanchi Ren, Tianchang Shen, Tianshi Cao, Kai He, Yifan Lu, Ruiyuan Gao, Enze Xie, Shiyi Lan, Jose M. Alvarez, Jun Gao, Sanja Fidler, Zian Wang, and Huan Ling. Chronoedit: Towards temporal reasoning for image editing and world simulation. *arXiv preprint arXiv:2510.04290*, 2025. 7

[80] Yang Wu, Kaihua Zhang, Jianjun Qian, Jin Xie, and Jian Yang. Text2lidar: Text-guided lidar point cloud generation via equirectangular transformer. In *ECCV*, 2024. 3

[81] Zijie Wu, Yaonan Wang, Mingtao Feng, He Xie, and Ajmal Mian. Sketch and text guided diffusion model for colored point cloud generation. In *ICCV*, 2023. 3

[82] Weiqing Xiao, Hao Huang, Chonghao Zhong, Yujie Lin, Nan Wang, Xiaoxue Chen, Zhaoxi Chen, Saining Zhang, Shuocheng Yang, Pierre Merriaux, et al. Simulate any radar: Attribute-controllable radar simulation via waveform parameter embedding. *arXiv preprint arXiv:2506.03134*, 2025. 3

[83] Enze Xie, Junsong Chen, Junyu Chen, Han Cai, Haotian Tang, Yujun Lin, Zhekai Zhang, Muyang Li, Ligeng Zhu, Yao Lu, et al. Sana: Efficient high-resolution image synthesis with linear diffusion transformers. *arXiv preprint arXiv:2410.10629*, 2024. 2, 4, 6, 14

[84] Yuwen Xiong, Wei-Chiu Ma, Jingkang Wang, and Raquel Urtasun. Learning compact representations for lidar completion and generation. In *CVPR*, 2023. 3

[85] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. In *ICCV*, 2019. 3

[86] Zhengqing Yun and Magdy F Iskander. Ray tracing for radio propagation modeling: Principles and applications. *IEEE access*, 2015. 3

[87] Maciej Zamorski, Maciej Zięba, Piotr Klukowski, Rafał Nowak, Karol Kurach, Wojciech Stokowiec, and Tomasz Trzciński. Adversarial autoencoders for compact representations of 3d point clouds. *Computer Vision and Image Understanding*, 2020. 3

[88] Yuchen Zhang, Nikhil Keetha, Chenwei Lyu, Bhuvan Jhamb, Yutian Chen, Yuheng Qiu, Jay Karhade, Shreyas Jha, Yaoyu Hu, Deva Ramanan, Sebastian Scherer, and Wenshan Wang. Ufm: A simple path towards unified dense correspondence with flow. In *arXiV*, 2025. 4, 5, 6

[89] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *ICCV*, 2021. 3

[90] Vlas Zyrianov, Xiyue Zhu, and Shenlong Wang. Learning to generate realistic lidar point clouds. In *ECCV*, 2022. 2, 3

RadarGen: Automotive Radar Point Cloud Generation from Cameras

Supplementary Material

This supplementary material includes evaluation metrics formulation, additional implementation details and additional experimental results.

Contents

A. Evaluation Metrics

This section provides the detailed formulations for the evaluation metrics introduced in the main paper.

Our evaluation is divided to the *Entire Area* ,which includes the whole point cloud in our range-of-interest (±50m), and the *Foreground*, which is the set of annotated bounding boxes with (1) camera visibility of more than 60%, (2) object class in {Car, Truck, Trailer}. Comparison is done between evaluated model predictions (synthetic points) and the ground truth points.

A.1. Chamfer Distance (CD)

Calculated as the two-way chamfer distance averaged across points and averaged by the two directions:

$$\mathrm{CD}(P_1, P_2) = \left(d(P_1, P_2) + d(P_2, P_1)\right)/2 \,,$$

where

$$d(P_1, P_2) = \frac{1}{|P_1|}\sum_{\mathbf{x}\in P_1}\min_{\mathbf{y}\in P_2}\|\mathbf{x}-\mathbf{y}\|_2 \,.$$

For CD-Loc we define P_1 and P_2 as the set of synthetic and ground truth locations $\{x_i^{syn}, y_i^{syn}\}$, $\{x_i^{gt}, y_i^{gt}\}$ respectively. For CD Full, we define P_1 and P_2 as the set of normalized locations and attributes Normalize$(\{x_i, y_i, r_i, d_i\})$, where each value is normalized to the range $[0, 1]$ to maintain a similar scale:

$$u_{\text{normalized}} = \frac{u - u_{\text{min}}}{u_{\text{max}} - u_{\text{min}}} \,.$$

This CD is defined only when there is at least one point in both P_1 and P_2 and in the *Foreground* it is only calculated on such bounding boxes. While most bounding boxes in our case contain points, this is not always guaranteed. Therefore, we present the Density Similarity metric in Sec. A.3 to verify that the similarity of point counts per object matches the ground truth. We report CD-Loc and CD-Full for the *Entire Area*, averaged across all point cloud pairs, as well as for the *Foreground*, over valid bounding boxes.

A.2. Point Cloud IoU@1m

We adopt the definition from [64], which computes the Intersection over Union (IoU) between two point clouds, P_1 and P_2, which are sets of locations $\{x_i^{syn}, y_i^{syn}\}$, $\{x_i^{gt}, y_i^{gt}\}$ respectively, using a matching distance threshold δ.

$$IoU = \frac{P \cdot R}{P + R - P \cdot R} \,,$$

where precision and recall

$$\begin{aligned} P &= \tfrac{1}{|P_1|}\textstyle\sum_{p_1\in P_1}\mathbb{I}\left[\min_{p_2\in P_2}\|p_2 - p_1\|_2 < \delta\right], \\ R &= \tfrac{1}{|P_2|}\textstyle\sum_{p_2\in P_2}\mathbb{I}\left[\min_{p_1\in P_1}\|p_1 - p_2\|_2 < \delta\right]. \end{aligned}$$

In our setting, we set $\delta = 1m$. We report IoU@1m for the *Entire Area*, averaged across all point cloud pairs.

A.3. Density Similarity

For the point cloud inside each bounding box we measure the difference in number of points compared to the ground truth points in that box. Define N, M as the number of points in point clouds P_1, P_2 respectively. Then the density similarity between P_1 and P_2 is defined as:

$$DS(P_1, P_2) = \begin{cases} 1 & \text{if } N = 0 \text{ and } M = 0 \\ \frac{\min(N,M)}{\max(N,M)} & \text{otherwise} \end{cases} .$$

We report Density Similarity for the *Foreground*, averaged across all bounding boxes.

A.4. Bounding Box Hit Rate

We consider the set of bounding boxes that contain at least one ground-truth point. The Hit Rate is the fraction of these bounding boxes that also contain at least one synthetic point.

$$\text{Hit Rate} = \frac{\text{Bounding boxes with synthetic points}}{\text{Bounding boxes with ground truth points}}\,.$$

We report Hit Rate for the *Foreground*, averaged across all bounding boxes.

A.5. Distance-Attribute (DA)

We propose the Distance-Attribute (DA) metric to jointly evaluate the accuracy of point locations and attributes. We define a "hit" only if a synthetic point falls within specific difference thresholds for spatial distance, RCS, and Doppler relative to a ground truth point. Unlike high-dimensional Chamfer Distance (CD-Full), DA enforces strict spatial locality, ensuring that spatially distant points with similar attributes are not matched. To avoid ambiguity arising from point ordering in greedy approaches, we solve a global assignment problem to identify the set of pairs satisfying all conditions. We define:

$$\begin{aligned} TP &= \text{\# of matched pairs},\\ FN &= \text{\# of unmatched GT points},\\ FP &= \text{\# of unmatched synthetic points}, \end{aligned}$$

from which Precision, Recall, and F1 are derived. In our setting, we select the thresholds: $\delta_{loc} = 1\text{m}$, $\delta_{RCS} = 8\text{dBsm}$, and $\delta_{Doppler} = 2.5\text{m/s}$. We report DA Recall, Precision, and F1 for the *Entire Area*, averaged across all point cloud pairs.

A.6. Maximum Mean Discrepancy (MMD)

To measure the distributional similarity between the synthetic point cloud P_1 and the ground truth P_2, we utilize Maximum Mean Discrepancy (MMD) [27]. We compute MMD independently for each point cloud attribute: location P_i^{Loc}, RCS P_i^{RCS} and Doppler P_i^{Doppler}. We employ a multi-scale Radial Basis Function (RBF) kernel defined as a sum of $K = 5$ Gaussians:

$$k(u, v) = \sum_{l=1}^{K} \exp\left(-\frac{\|u - v\|^2}{h_l}\right).$$

The kernel bandwidths are set as $h_l = h_{\text{base}} \cdot 2^{l-3}$, where the base bandwidth h_{base} is calculated as the mean squared Euclidean distance between all distinct pairs in the joint set $P_1^{\text{attr}} \cup P_2^{\text{attr}}$ for a given attribute. The final MMD metric is computed as the biased estimator over the resulting kernel matrix.

Entire Area. For each pair of synthetic and ground truth point clouds, we report MMD separately on location, RCS, and Doppler.

Foreground. For each class {Car, Truck, Trailer}, we aggregate points from annotated bounding boxes after transforming them into a canonical coordinate system via centering and rotation alignment. We report the MMD between the aggregated synthetic and ground truth sets for each class, calculated separately for location, RCS, and Doppler.

B. Additional Details

B.1. Additional details on BEV representation

Radar maps. Before converting radar maps to latent representation we require an image that would serve as the encoder input. Maps M_p, M_r, M_d possess values in the following ranges:

$$\begin{aligned} M_p &\in [-50\text{m}, 50\text{m}]\,,\\ M_r &\in [-20\text{dBsm}, 66\text{dBsm}]\,,\\ M_p &\in [-120\text{m/s}, 120\text{m/s}]\,. \end{aligned}$$

We use these values to normalize the maps to the range $[0, 255]$.

BEV scenes conditioning. When converting images to point maps and subsequently to a 512×512 BEV grid, we filter the point maps to reduce input noise. Specifically, we discard points corresponding to object edges and sky regions. We also remove points with a predicted height greater than 5m to filter out overhead structures such as bridges and trees. Finally, if multiple points occupy a single grid cell, we retain only the point with the maximum height. Fig. 11 shows visualizations of the BEV conditioning maps.

Explainability. A key advantage of BEV scene conditioning is interpretability; it clarifies exactly what information is available to the model, allowing us to anticipate and explain the output radar maps. This contrasts with models conditioned solely on camera images, where the model's internal awareness of object existence, classification, and location remains opaque.

B.2. Additional details on training

We train *RadarGen* for 2 days on 8 L40 (48GB) GPUs, totaling 65k steps. We initialize the model using SANA's [83] pre-trained 600M-parameter weights at 512×512 resolution, utilizing SANA AE v1.1. We fine-tune the model using a batch size of 16 per GPU, seed 42, gradient accumulation of 2, bf16 mixed precision, and a learning rate of 10^{-4}. Additionally, we apply conditioning dropout with a probability of 10%, replacing the condition with a zero tensor. To improve training efficiency, we pre-compute and store the required BEV images during a preprocessing step.

B.3. Additional details on inference

During inference of our diffusion model we use 20 sampling steps, a null prompt, and no guidance scale. For reproducibility, the initial seed is set to 42. During the sparse point cloud recovery, we deconvolve the radar maps via an IRL1 solver with FISTA. We utilize the following hyperparameters: $\lambda = 0.0018$, 300 FISTA iterations, 5 IRL1 iterations, and a threshold of 0.1 for the sparse map $\mathcal{P}'_{xy}$.

Inference time. The total inference time for a single timestep t on a single L40 GPU is approximately 10.5 seconds. This decomposes into 9 seconds for BEV conditioning map creation, 1 second for diffusion inference, and 0.5 seconds for point cloud recovery.

B.4. Additional details on the baseline model

We employ RGB2Point [40] as our multi-view image-to-point cloud baseline. This model utilizes a pre-trained ViT to extract features from an arbitrary number of input views and maps them to a point set. We adapt the architecture to predict RCS and Doppler attributes by increasing the point cloud dimension from 3 to 5. Additionally, we increase the number of attention heads to 8, the intermediate linear dimension to 2048, and the feedforward dimensionality to 4096. Input images are resized to 224×224 and normalized following the original method. We retain the Chamfer Distance (CD) loss, computed across all 5 channels. The output is a fixed set of 1024 points. The model was trained with a batch size of 8 and a learning rate of 10^{-5} on 8 L40 (48GB) GPUs for 18 hours. We observed that the model failed to converge when using the normalized "CD Full" objective; therefore, we opted not to normalize the values and minimized the Chamfer Distance on the full unnormalized vector.

B.5. Additional details on VoxelNeXt

We adapt VoxelNeXt [12] for radar input, training for 77k steps (9 hours) on 8 L40 (48GB) GPUs. We use a batch size of 32 per GPU and apply data augmentations including flipping, rotation, translation, and scaling. The point cloud range is set to $[-50\text{m}, 50\text{m}]$.

B.6. Additional details on direct multi-view conditioning

To condition radar maps generation on multi-view camera images, we apply the same shared self-attention mechanism used for multi-map denoising. The denoising process involves 11 latent tensors: 3 for the radar maps, 4 for camera images at time t, and 4 for camera images at time $t+1$. Each image is transformed to the model's required 512×512 resolution by padding and resizing. Each image is then encoded to into a latent tensor and concatenated along the features dimension with its corresponding Plücker coordinates (adjusted for padding and relative to the ego vehicle at time t) and a unique modality indicator. Similarly, the radar map latents are concatenated with a zero tensor to match the feature dimensionality, along with a unique modality indicator. We train this model for 65k steps on 8 L40 (48GB) GPUs, which takes 9 days, using the same hyperparameters as *RadarGen*.

C. Additional Results

C.1. Additional qualitative results

We provide additional qualitative results for highway and rural environments (Fig. 7), as well as urban areas (Fig. 8). We also illustrate the effect of different random seeds (Fig. 9). Finally, we include supplementary videos demonstrating scene generation.

C.2. Additional radar PCL recovery results

Fig. 10 compares the Random, Peak, and Deconvolution sparse point cloud recovery methods on generated scenes. The Random sampling method treats the point density map as a probability distribution, resulting in an inconsistent point distribution. This causes gaps in the data; for instance, in the third row, the moving car is not captured, while other areas exhibit excessive density. The Peak method, which selects the local maximum within a 3×3 window, results in a sparse point cloud that captures all relevant regions but lacks sufficient density. The Deconvolution method combines the strengths of both approaches, covering all relevant regions while ensuring sufficient density where required.

C.3. Additional information on limitations

Fig. 12 illustrates a low-light night scene where the underlying foundation models struggle to accurately recognize vehicles and estimate their velocities. Although *RadarGen* was not trained on such scenarios, we present a qualitative comparison against the ground truth radar.

C.4. Additional detection results

Tab. 3 details the detection model's performance on ground truth data compared to synthetic data produced by *RadarGen* and the baseline. We evaluate detection on the Car, Truck, and Trailer classes.

Table 3. **Detection metrics comparison.** Evaluation of a trained detector on GT versus generated samples from *RadarGen* and Baseline.

Method	mAP ↑	mATE ↓	mASE ↓	mAOE ↓	mAVE ↓	mAAE ↓	NDS ↑
GT	0.38	0.54	0.18	0.11	1.88	0.24	0.48
RadarGen	0.11	0.91	0.20	0.21	4.04	0.19	0.30
Baseline	0.00	1.00	1.00	1.00	1.00	1.00	0.00

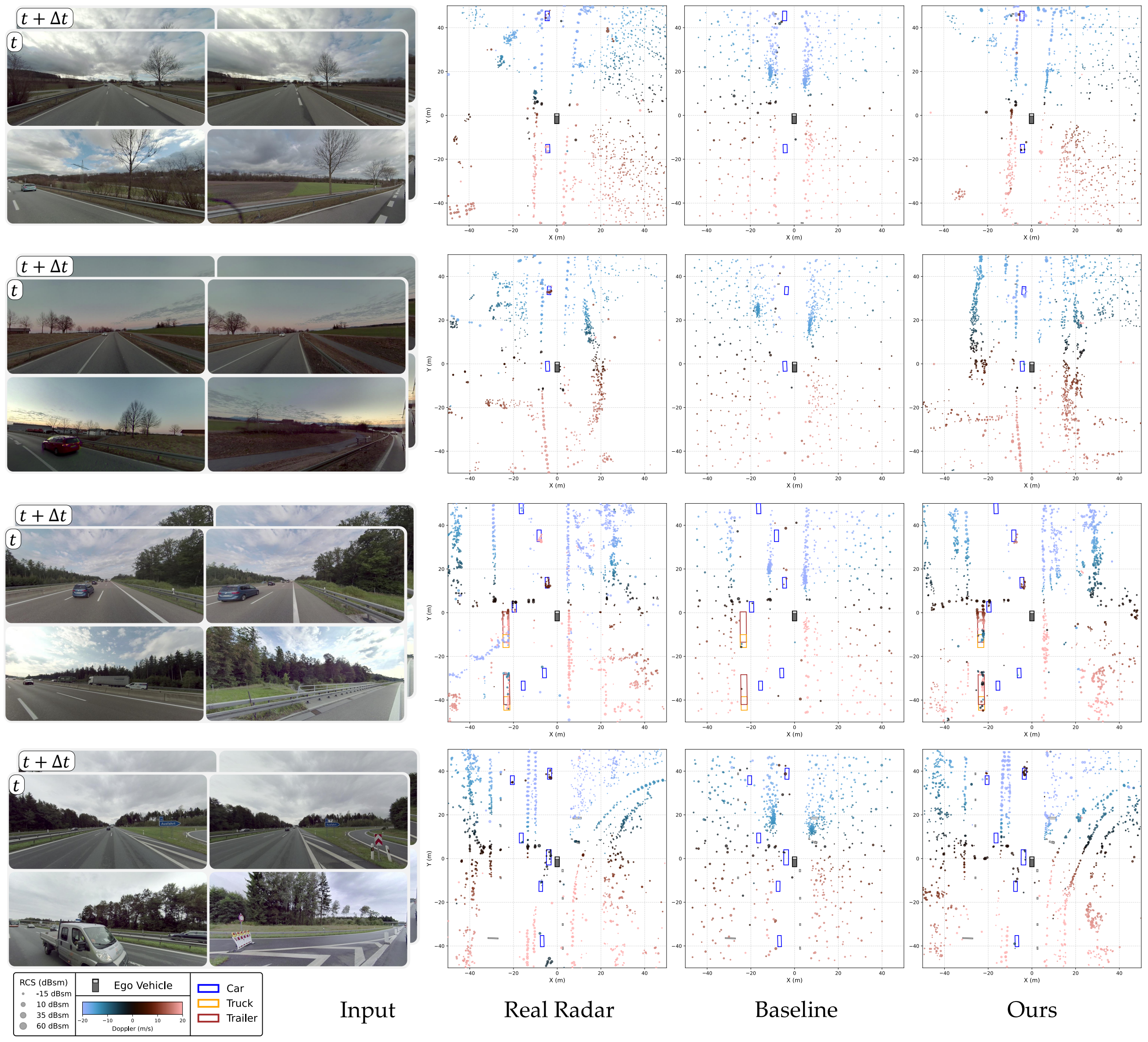

Figure 7. **Additional qualitative results.** Additional demonstration of *RadarGen* compared to the baseline and ground truth in highway and rural scenarios. Our model generates point clouds with higher geometric and attribute fidelity to the ground truth compared to the baseline. *RadarGen* uses inputs t and $t + \Delta t$, while the baseline uses only t. Ground truth bounding boxes are highlighted in color.

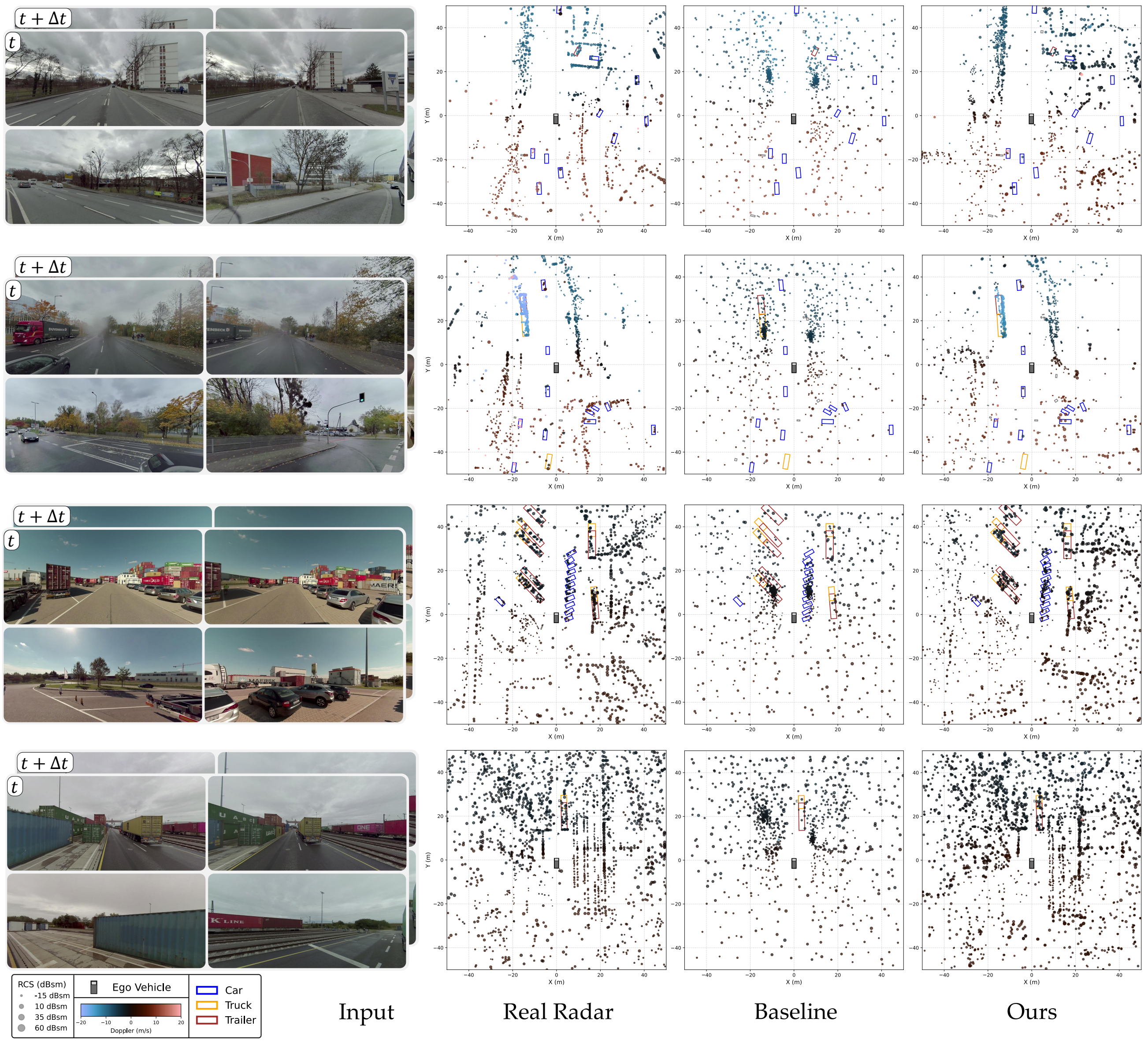

Figure 8. **Additional qualitative results.** Additional demonstration of *RadarGen* compared to the baseline and ground truth in urban environments. Our model generates point clouds with higher geometric and attribute fidelity to the ground truth compared to the baseline. *RadarGen* uses inputs t and $t + \Delta t$, while the baseline uses only t. Ground truth bounding boxes are highlighted in color.

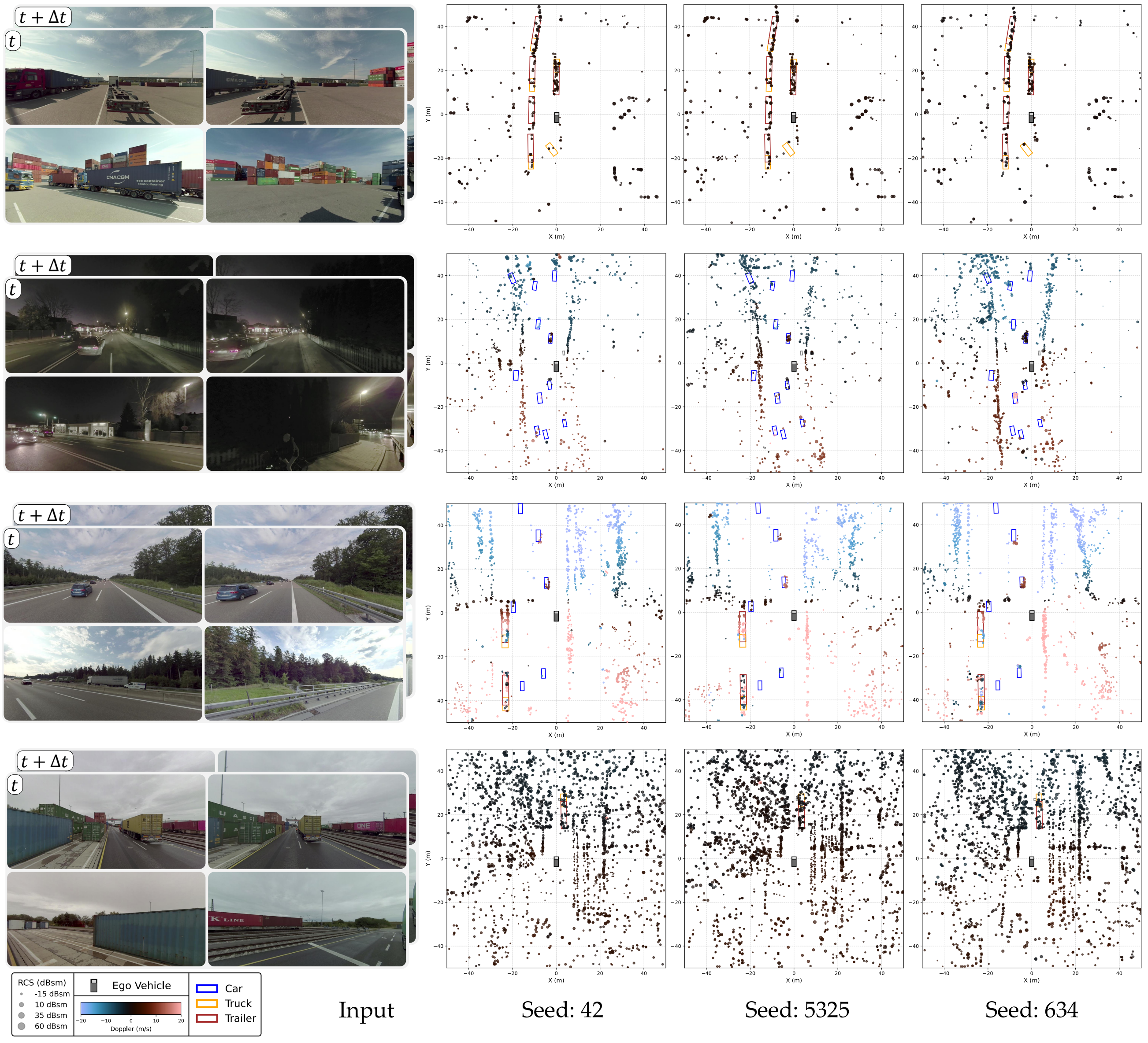

Figure 9. **Additional seeds.** Our model can generate multiple sets of point clouds for a single scene by replacing the diffusion process seed. *RadarGen* uses inputs t and $t + \Delta t$. Ground truth bounding boxes are highlighted in color.

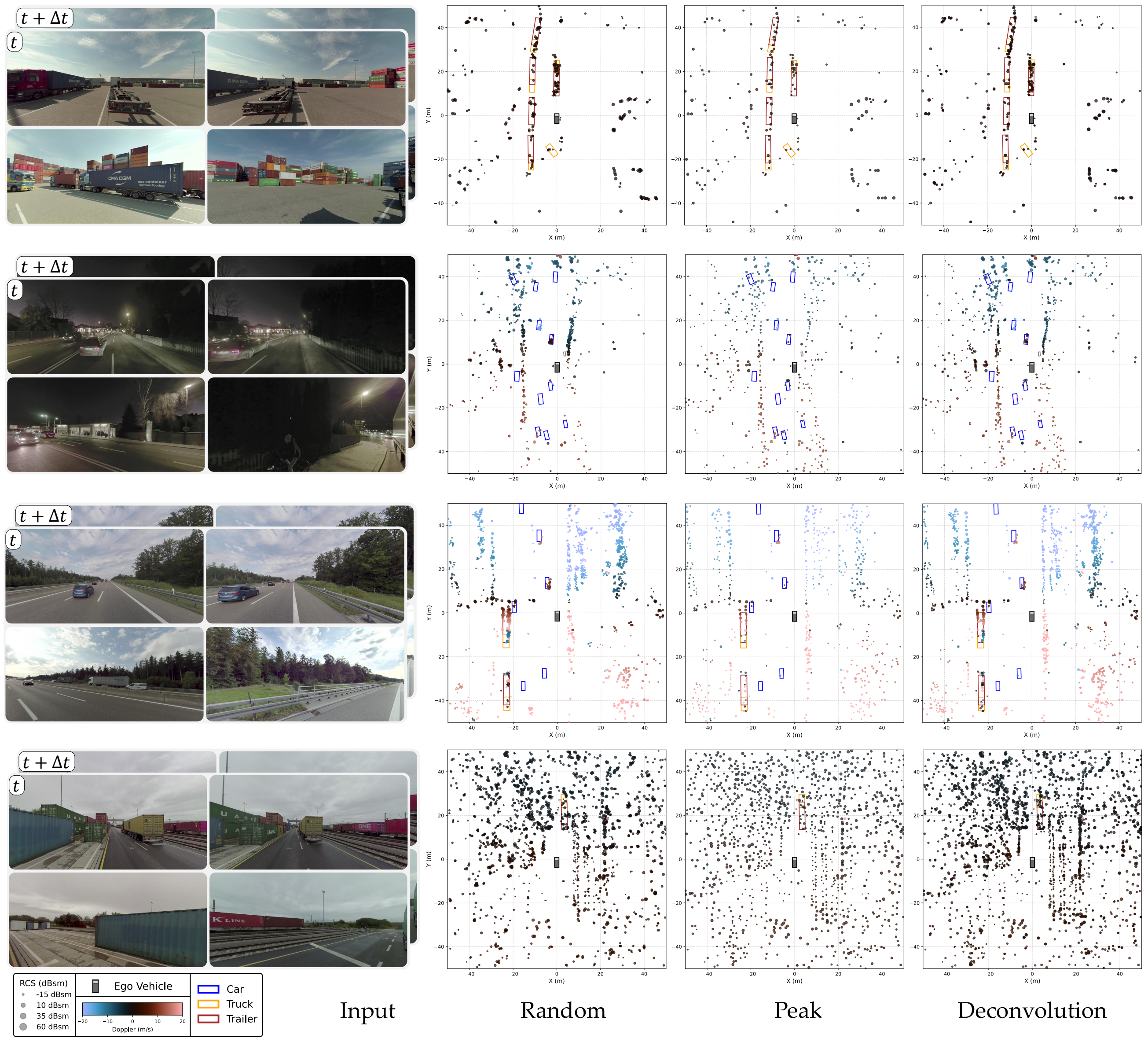

Figure 10. **Recovery methods.** Comparison of Random, Peak, and Deconvolution sparse point cloud recovery methods. Random sampling exhibits inconsistent density characterized by clustering and empty regions. Peak recovery fills the space uniformly but suffers from low density. Our Deconvolution method achieves coverage while maintaining density where necessary. *RadarGen* uses inputs t and $t + \Delta t$. Ground truth bounding boxes are highlighted in color.

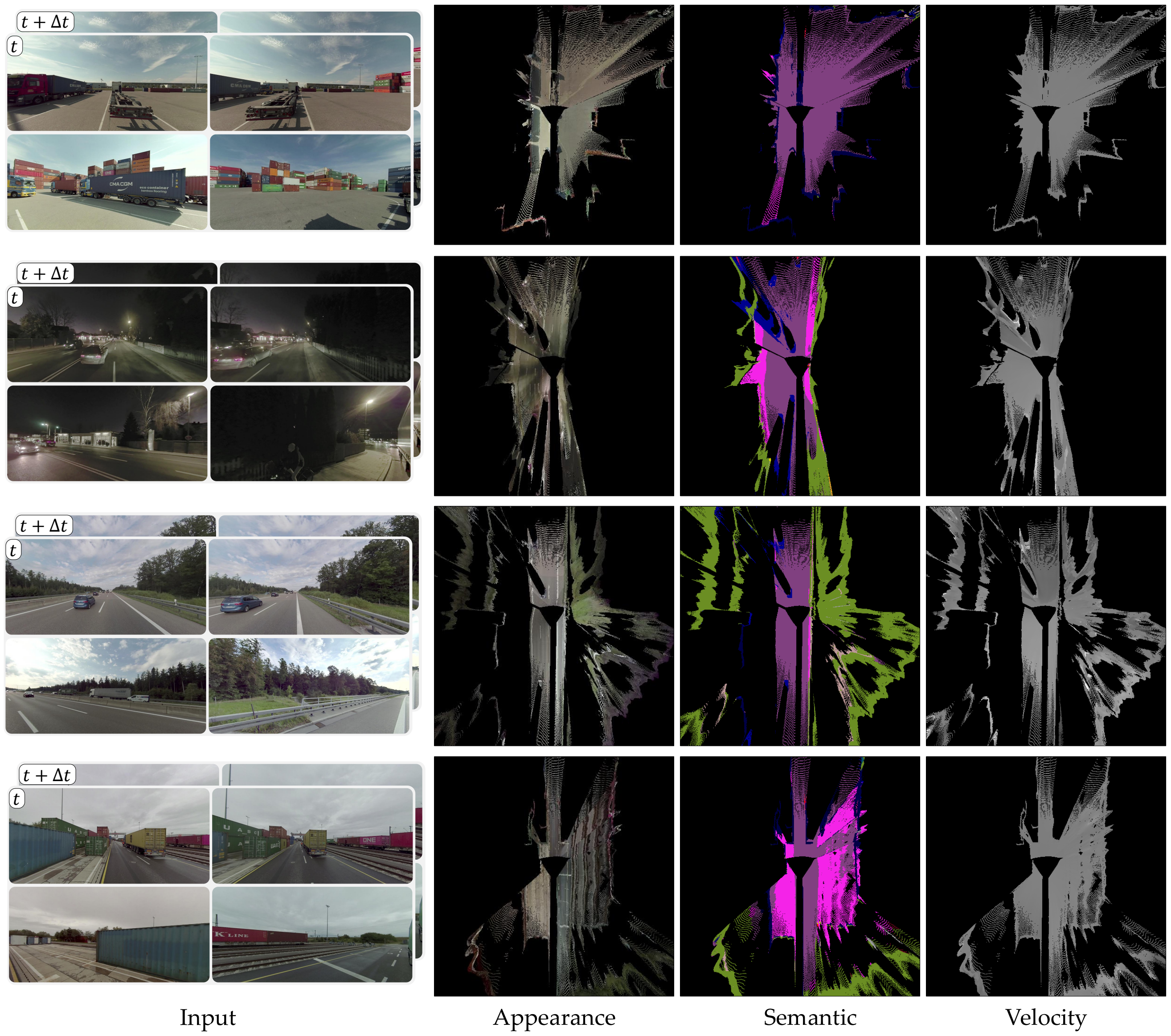

Figure 11. **BEV conditioning maps.** Visualization of the BEV appearance, semantic, and relative radial velocity maps produced from inputs at times t and $t+\Delta t$. The appearance map retains the camera image colors. Semantic classes are color-coded as: 🟪 Road, 🟪 Sidewalk, ⬛ Building, 🟩 Vegetation, 🟦 Car, and 🟥 Person. For the velocity map, lighter colors indicate positive velocity while darker colors indicate negative velocity.

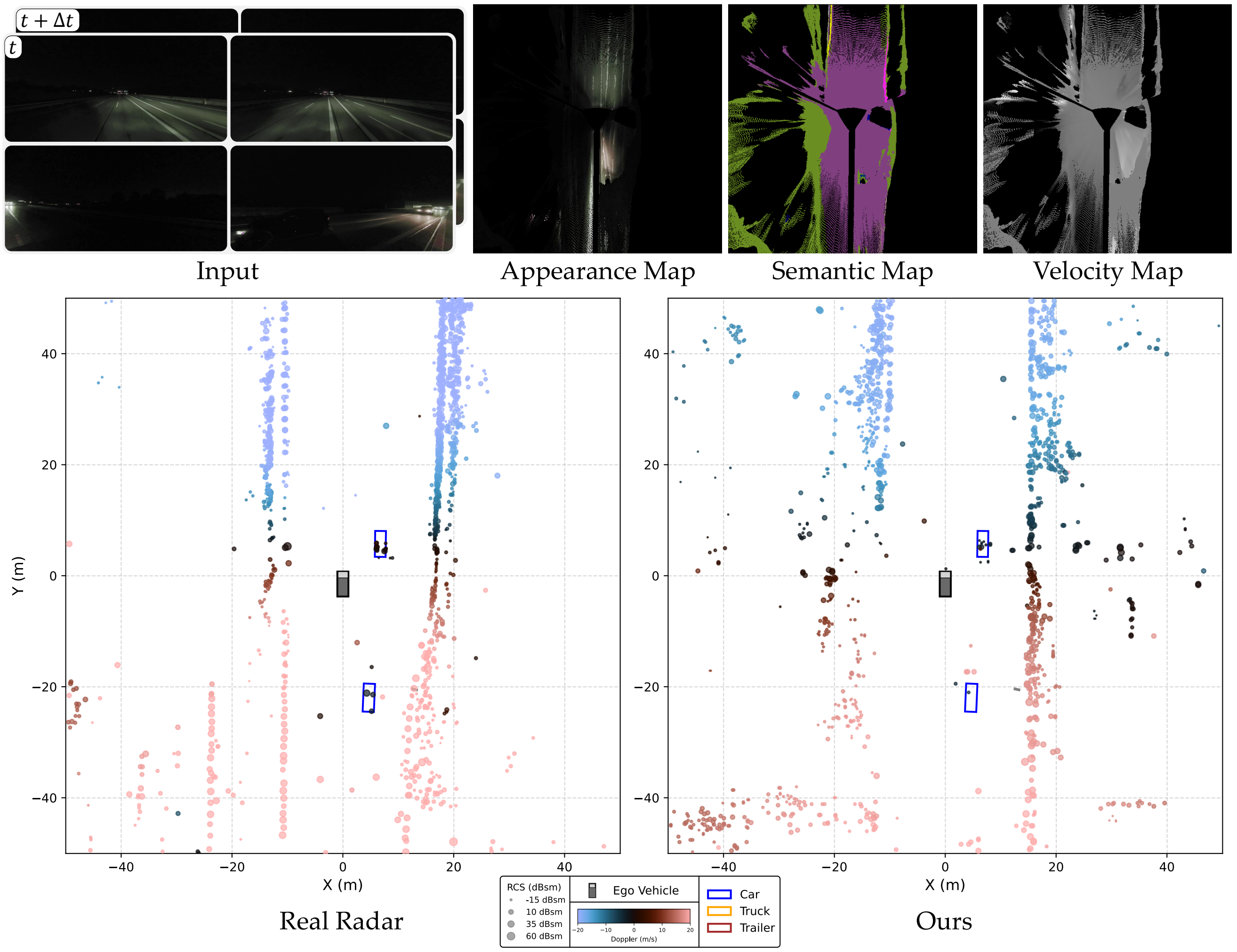

Figure 12. **Qualitative analysis of limitations.** Visual comparison of *RadarGen* against the ground truth radar in a low-light night scene. In this setting, the underlying foundation models struggle to accurately recognize vehicles and estimate velocities. *RadarGen* was not trained on such scenarios. Ground truth bounding boxes are highlighted in color.