

## 1. Supplementary Videos

Please refer to the attached supplementary videos for results. We show comparison between *RadarSim*- Radar reflectance rendered with radar occupancy (**left column**) and radar-only baselines DART [6](**middle column**) and Radarfields[2](**right column**). Observe that compared to the baseline, our reconstruction quality shown in depth map at the bottom is significantly better, while being able to accurately model radar reflectivity. Specifically, strong reflectance is visible for radar signals:

- at inset corners that act as retro-reflectors where all transmitted rays are reflected due to bouncing at corners, such as bottom of car, inside windows, wall/ceiling intersections, light fixture on ceilings etc.
- where view direction aligns with surface normal
- metallic material in general

We also show synthesized range-azimuth view in 128 azimuth directions, achieving super-resolution of the input radar data which only contains 8 directions from 8 antenna measurement. Notice our synthesized range-azimuth rendering is much sharper than DART[6], while preserving radar reflectivity.

## 2. Dataset

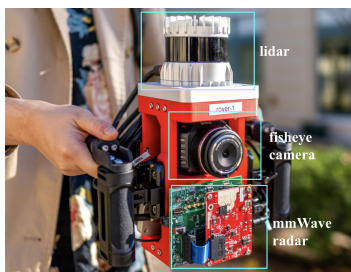


Figure 1. Image of our hand-held data collection rig with three key components labeled.

### 2.1. Data capture rig and pose processing

We build a hand-held rig for data collection with time-synced mmWave radar, fisheye camera, and lidar collecting data at 30 fps, 30 fps and 10 fps respectively.

For each sequence, we run COLMAP to get camera poses  $\mathbf{A}_c$  (can be broken down into rotation  $\mathbf{R}_c$  and position  $\mathbf{x}_c$ ). Since coordinate systems of camera and radar are calibrated, we can easily convert camera pose to radar pose  $\mathbf{A}_r$  (or  $\mathbf{R}_r$  and  $\mathbf{x}_r$ ) by interpolating into the sequence of camera poses with synced-timestamps followed with a transformation. We use the proposed scale estimation approach mentioned in Section 3.5 of the main paper to obtain scale of the scene for estimating ego-velocity required for our system.

### 2.2. Multimodal pose refinement

While COLMAP provides fairly accurate poses, radar ray tracing requires accurate **velocity**, so we design a pose and velocity refinement module by learning a per-frame pose and velocity offset  $\Delta\mathbf{x}_c, \Delta\mathbf{R}_c, \Delta\mathbf{v}_c$  and regularized through:

1. L2 regularization on the offsets:  $L_{regp} = \|\Delta\mathbf{x}_c\|_2^2 + \|\Delta\mathbf{R}_c\|_2^2 + \|\Delta\mathbf{v}_c\|_2^2$
2. L2 loss that enforces velocity to be close to derivative of position  $L_{regv} = \|d(\mathbf{x}_c)/dt - (\mathbf{v}_c + \Delta\mathbf{v}_c)\|_2^2$
3. L2 regularization on acceleration  $L_{rega} = \|d(\Delta\mathbf{v}_c + \mathbf{v}_c)/dt\|_2^2$
4. kinematic loss:  $L_{regk} = \|d_{window}(\mathbf{x}_c + \Delta\mathbf{x}_c) - d_{window}(\int \Delta\mathbf{v}_c + \mathbf{v}_c dt)\|_2^2$

Note that it is possible to derive optimized velocity from optimized positions ( $\mathbf{x} + \Delta\mathbf{x}$ ), but we empirically found it to be more stable to keep a different set of velocity offset and position offset and have them loosely connected through regularization. Such pose optimization scheme is shared across camera and radar as radar pose and velocity can be interpolated from camera poses and velocities. We show in Table 3 that our proposed pose refinement method improves on original DART[6] without pose refinement.

### 2.3. Evaluation traces break down

We show lidar map of the scene, trajectory and a RGB view of our 8 evaluation traces in Figure 2. Number of images and radar frames range from 2000 to 3000. Image size used for training is 960x540 px.

### 2.4. Existing multimodal camera-radar dataset

We include a summary of existing multimodal datasets that contain raw radar measurement in Table 1 in the main paper and also attached here that we further elaborate on. There is a lack of multimodal radar-camera dataset that contains raw single-chip radar measurement for scene reconstruction, which requires overlapping, and view-direction varying scene content across multiple measurements. Automotive datasets, such as RADial [4] and RaDICAL [8], offer limited variation in viewpoints and sensor height, making it difficult to use for evaluating novel view synthesis. RADial [4] also uses Cascaded Radar with a complex modulation function, which makes modelling radar rendering equation difficult for inverse rendering. Coloradar [7] does capture indoor scenes that can be used for reconstruction, but uses high-resolution radar which is also beyond our sin-radar focus. Most public RGB-Radar datasets, such as NuScenes [3], provide only *post-processed* CFAR data—not the *raw* measurements that we use. Hence we collect our own multimodal dataset and evaluate performance of our approach and baselines.

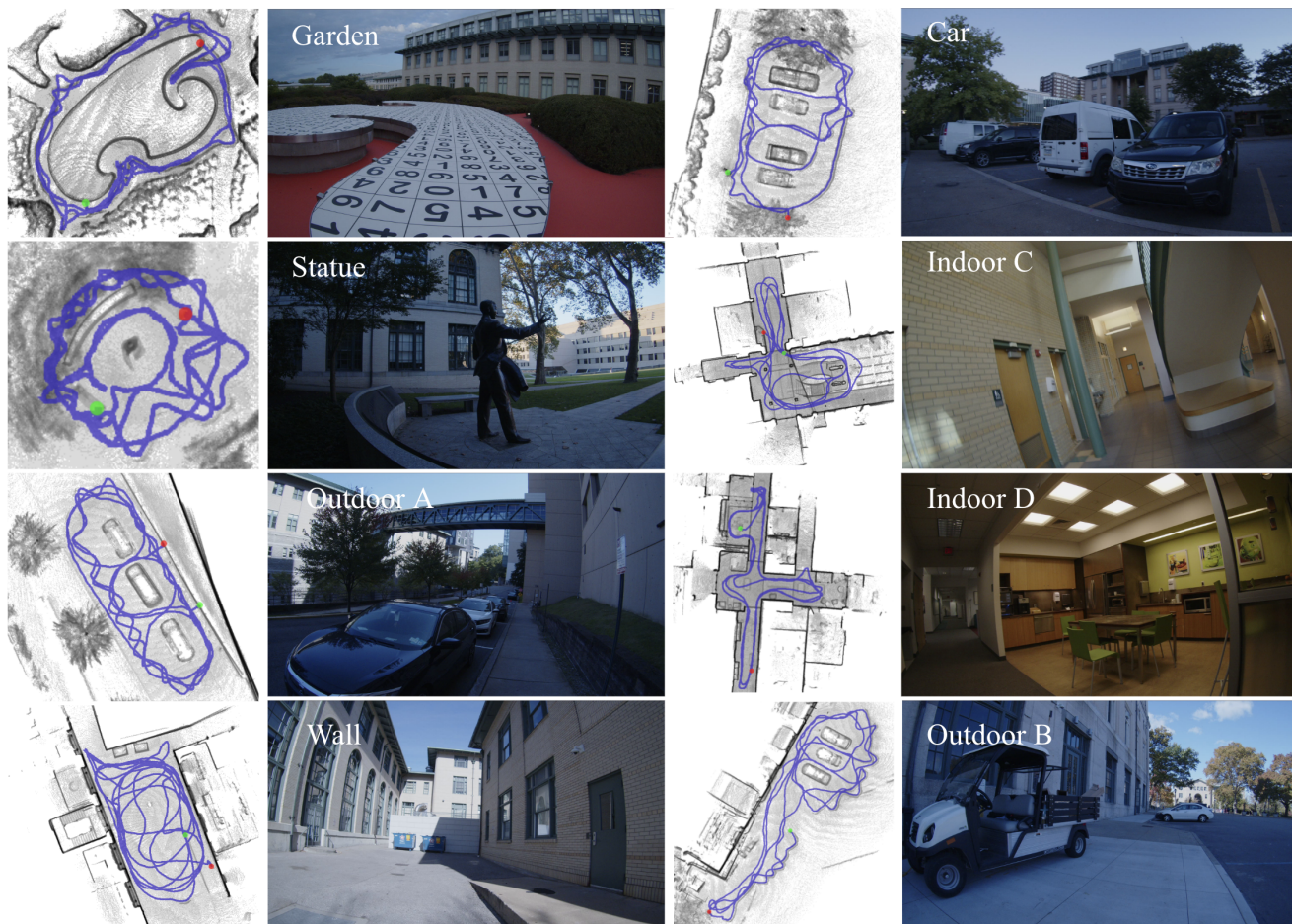


Figure 2. Visualization of the 8 indoor and outdoor scenes we collect and evaluate on in our experiments, in lidar map (**left**) and a sample RGB image (**right**).

Dataset	Radar Type	Raw Data	View-direction Varying	Other Sensors
RadarSim (Ours)	<b>Low Res</b>	<b>Yes</b>	<b>Yes</b>	Lidar, Camera, IMU
RADDet [14]	<b>Low Res</b>	<b>Yes</b>	No	Camera
RADial [11]	High Res	<b>Yes</b>	No	Lidar, Camera, GPS
K-radar [10]	High Res	<b>Yes</b>	No	Lidar, Camera, IMU, GPS
Coloradar [7]	High Res	<b>Yes</b>	<b>Yes</b>	Lidar, IMU
RaDICAL [9]	<b>Low Res</b>	<b>Yes</b>	No	Depth Camera, IMU

Table 1. **Comparison with other mmWave radar datasets with raw data.** We capture a dataset using a low-resolution single-chip radar and cover scene content from multiple views directions and positions.

### 3. Method details

We elaborate here on technical details omitted in the main paper. See figure 3 for detailed architecture diagram.

#### 3.1. Modelling radar view dependence with BRDF bases

View dependence of Radar reflectance can be broken into 2 scenarios, surface normal dependent reflectance and surface normal independent reflectance. The latter includes

reflectance from retro-reflectance structures such as corners, bottom of car etc. ( where rays always bounce back in a particular direction) and inter-reflections. To model these 2 scenarios, we input to Radar MLP with BRDF bases to model normal dependent view dependence, and spherical harmonics encoded view directions to model normal independent view dependence. Geometry code is also input to the Radar MLP as both types of reflections are spatially varying.

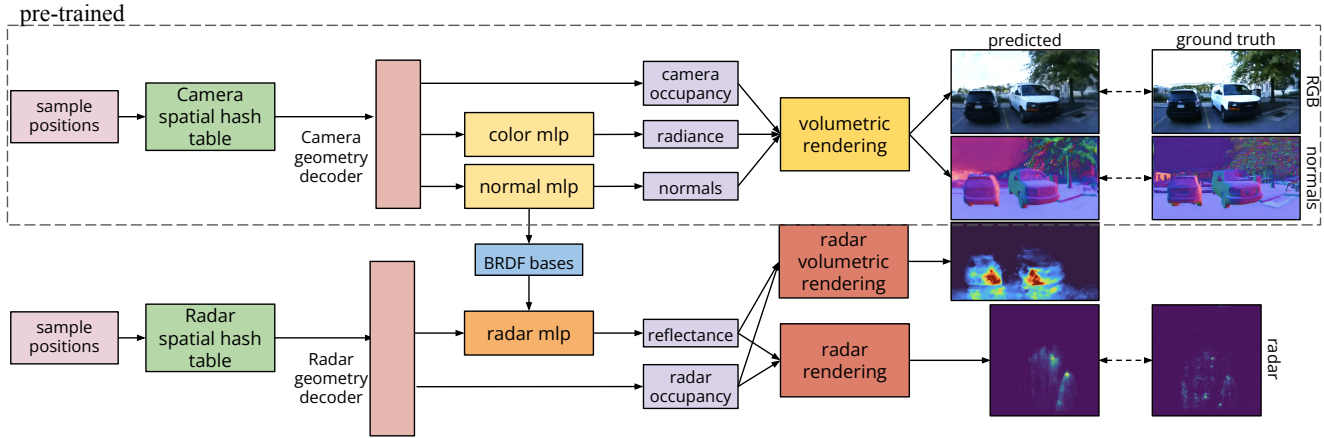


Figure 3. Architecture diagram of our framework.

### 3.2. Training details

Following DART [6], we process radar raw data into Range-Doppler-Azimuth frames with dimension size 128, 128, 8. At each training iteration, a batch of radar Doppler columns  $Y_r$  are sampled to form radar rays. Radar rays are sampled on a cone with directions determined by velocity of the sensor at the current frame, and apex angle determined by the dot product between speed and Doppler value of the sampled column.

**Radar rendering.** Radar ray batch is sampled using a fine-tuned radar proposal sampler which outputs different sampling weights from camera. The hash table and density MLP corresponding to  $f_{geo_r}$  are initialized with pre-trained  $f_{geo_c}$  and optimized using radar measurements. They are queried to obtain radar occupancy and a geometry code, which is decoded and fed to the Radar MLP along with SH encoded view directions and our proposed BRDF bases to decode into radar reflectance. Reflectance and radar occupancy are assigned to each range bin as discussed in Sec 3.3 in the main paper, and rendered with DART rendering equation to synthesize the input Doppler column  $\hat{Y}_r$  [6].

### 3.3. Loss Functions

RGB model is supervised with losses used in Nerfacto. Our model is supervised with  $L_1$  reconstruction loss and SSIM loss for radar measurement, binary cross entropy loss between radar and camera occupancy to constrain radar geometry to be close to camera geometry, as well as auxiliary losses for pose regularization (mentioned in Section 2.2), proposal sampling network and normals. We list the loss functions here:

#### Reconstruction loss

$$L_r = \|Y_r - \hat{Y}_r\|_1 \quad (1)$$

#### SSIM loss

$$L_{ssim} = 1 - SSIM(Y_r, \hat{Y}_r) \quad (2)$$

#### Geometry consistent loss

$$L_{bce} = -[\alpha_c \log(\alpha_r) + (1 - \alpha_c) \log(1 - \alpha_r)] \quad (3)$$

**Interlevel Losses** We fine-tune the proposal sampler for radar with a proposal loss  $L_{prop}(\mathbf{t}, \mathbf{w}, \hat{\mathbf{t}}, \hat{\mathbf{w}})$  discussed in [1] to encourage histogram of rendering weights  $\hat{\mathbf{w}}$  queried from the proposal network at samples  $\hat{\mathbf{t}}$  to match the rendering weight  $\mathbf{w}$  of the geometry field at a set of different sample positions  $\mathbf{t}$ . Specifically,

$$L_{prop}(\mathbf{t}, \mathbf{w}, \hat{\mathbf{t}}, \hat{\mathbf{w}}) = \sum_i \frac{1}{w_i} \max(0, w_i - \text{bound}(\hat{\mathbf{t}}, \hat{\mathbf{w}}, T_i)) \quad (4)$$

where  $\text{bound}(\hat{\mathbf{t}}, \hat{\mathbf{w}}, T_i)$  is the sum of proposal weights  $\hat{\mathbf{w}}$  in interval  $i$ . This loss function penalizes the proposal weights that under-estimates the rendering weight distribution from geometry field. Please refer to [1] for details about this loss function. We apply this loss to radar as  $L_{prop_r}$ , to enforce the proposal network to generate sampling weights that focus on radar geometry. Radar rendering weights are computed by

$$w_r = \alpha_r(\mathbf{t}_i) \prod_{j < i} (1 - \alpha_r(\mathbf{t}_j)) \quad (5)$$

**Normal Supervision Losses** We supervise our normal prediction MLP with pseudo ground truth normal  $\mathbf{n}_{gt}$  from



a monocular normal estimator [5] by first converting it to world coordinate using camera extrinsics  $\mathbf{A}_c$ :

$$L_{norm} = \|\mathbf{n} - \mathbf{A}_c \mathbf{n}_{gt}\|_2^2 \quad (6)$$

Our combined loss is

$$L = \lambda_r L_r + \lambda_{bce} L_{bce} + \lambda_{ssim} L_{ssim} + \lambda_{prop_r} L_{prop_r} + \lambda_{norm} L_{norm} + \lambda_{norm_g} L_{norm_g} + \lambda_{norm_o} L_{norm_o} + \lambda_{regp} L_{regp} + \lambda_{regv} L_{regv} + \lambda_{rega} L_{rega} + \lambda_{regk} L_{regk} \quad (7)$$

In this equation,  $L_{norm_g} + L_{norm_o}$  are adapted from [13] to guide the predicted normal with gradient direction of the camera density field, and encourage normals to point outward from a surface. We use  $\lambda_r = 1e^{-3}$  for outdoor scenes and  $\lambda_r = 1e^{-4}$  for indoor scenes where abundance of multi-path reflections result in overall high reflectance in the scenes. We choose  $\lambda_{bce} = 0.01$  and  $\lambda_{ssim} = 0.01$ . We choose  $\lambda_{norm} = 0.1$  to obtain strong supervision from ground truth normal. We follow default values in Nerfstudio and use  $\lambda_{prop_r} = 1$ ,  $\lambda_{norm_g} = 1e^{-3}$ ,  $\lambda_{norm_o} = 1e^{-4}$ . For pose refinement, we use  $\lambda_{regp} = 1e^{-3}$ ,  $\lambda_{regv} = 1$ ,  $\lambda_{rega} = 5e^{-3}$ ,  $\lambda_{regk} = 1$ .

### 3.4. Implementation Details

We implement our pipeline using Pytorch in Nerfstudio [12] based on Nerfacto-big [12]. For training we used a single 24 GB Nvidia Rtx3090 GPU. Each sequence is trained on RGB images first for 50k iterations and fine-tuned on radar data for 30k iterations. Training time is around 2 hours. Learning rate for the radar model is  $1e^{-2}$  annealed to  $1e^{-4}$  after 30k steps, and for pose refinement is  $1e^{-3}$  annealed to  $1e^{-4}$  after 5k steps. We list model hyperparameters in Table 2.

## 4. Experiment details and additional result

### 4.1. Evaluation metric and denoising procedure

We calculate PSNR and SSIM values between *RadarSim* and ground truth for evaluation and comparison against baselines. Because most of the radar frame consists of noise, we design a denoising procedure by finding the noise threshold for each dataset. The noise threshold is calculated by fitting a chi-square distribution to the empty Doppler columns of each dataset (where speed is smaller than the Doppler values) and take a p-value of 0.01 of the noise distribution as the noise threshold. During evaluation, ground truth and synthesized range-Doppler frames are clipped at 0.01 and 99.99 percentile of the ground truth over the entire dataset and normalized to 0 and 1 to calculate SSIM and PSNR. Areas where the ground truth frames are below this threshold are considered to be noise and ignored during both PSNR and SSIM calculation.

### 4.2. Description of baselines

We run a pytorch version of DART [6] using code provided by the authors. The parameter for the hashtable and geometry decoder is set to match the size of *RadarSim*. We use the same set of poses for Radar for the baseline and our approach, where they are time-interpolated from COLMAP-derived camera poses. All comparisons included in the main paper with DART use additional pose refinement described in Section 2 for fair comparison with *RadarSim*. We include comparison with DART without pose refinement in Table 3. It can be observed that our proposed pose refinement scheme improves DART’s novel view synthesis quality as the model is less prone to overfitting to the inaccuracies in poses. We also implement RadarFields [2] as an additional radar-only baseline. Since this approach is designed for high resolution radar, it only uses range-azimuth. To compare with our method, we train it on range-azimuth slices of the input data only and evaluate on range-doppler-azimuth synthesis quality. We also include result in Table 3. Although this method includes pose refinement, its performance is inferior to that of DART with pose refinement, suggesting the importance of leveraging Doppler for low resolution radar.

### 4.3. Additional application: geometry improvement for textureless region

We further illustrate the effectiveness of our BRDF encoding for modelling radar reflectance in a multimodal training framework where a single geometry field is optimized to represent radar and camera and trained simultaneous using radar and camera reconstruction losses. Without using monocular normal or depth priors, RGB-only reconstruction fails at identifying the true geometry for textureless regions. In a multimodal training setting, since our BRDF encoding effectively models radar view-dependence using predicted normals from the shared geometry, information from radar measurement can be propagated to the geometry and reconstruct textureless regions correctly as shown in figure 4. Comparison with other RGB-based or multimodal-based methods in improving geometry quality is left to future work.



Figure 4. Normal-dependent BRDF encoding allows *RadarSim* to improve reconstruction of textureless regions in a joint training setting, without using monocular-predicted normals.



Model	Configuration	Value
SH	degree	25
BRDF bases	number	11
Proposal		
Hash	# of levels	5, 5
Encoding level 0,1		
	Hash table size	$2^{17}$ , $2^{17}$
	# of feature dim. per entry	2,2
	Coarse resolution	16,16
	Fine resolution	128, 256
	Decoder feature dim	16,16
	Number of layer	2,2
	# of ray samples	512, 256
Hash	# of levels	16
Encoding		
	Hash table size	$2^{21}$
	# of feature dim. per entry	2
	Coarse resolution	16
	Fine resolution	2048
	# of ray samples	64
Density	# of hidden layer	2
MLP		
	# of neuron per layer	128
	Output activation	Exp
	Density feature dim	15
Radar MLP	# of hidden layer	2
	# of neuron per layer	128
	Output activation	None
Color MLP	# of hidden layer	2
	# of neuron per layer	128
	Output activation	Sigmoid
Normal MLP	# of hidden layer	2
	# of neuron per layer	64
	Output activation	None

Table 2. List of hyperparameters used in our architecture.

	Car		Garden		Statue		Wall		Outdoor A		Outdoor B		Indoor C		Indoor D		Aggregated	
	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR
CFAR	0.694	24.9	0.415	24.1	0.711	17.9	0.772	27.1	0.502	20.2	0.734	25.3	0.774	28.0	0.770	28.2	0.671	24.5
Lidar	0.782	29.4	0.470	26.7	0.830	28.9	0.796	27.9	0.595	26.8	0.782	28.6	0.805	29.1	0.809	29.2	0.734	28.3
Nearest Neighbor	0.733	26.5	0.592	24.6	0.798	26.2	0.810	24.6	0.580	22.9	0.726	24.3	0.782	26.6	0.785	27.3	0.726	25.4
RadarFields [2]	0.831	29.3	0.603	27.0	0.869	29.0	0.781	27.1	0.667	24.4	0.773	27.5	0.814	28.6	0.831	29.3	0.771	27.8
DART [6] without pose refinement	0.832	29.8	0.616	27.1	0.866	29.1	0.809	27.0	0.691	26.3	0.797	27.1	0.817	28.5	0.846	29.3	0.784	28.0
DART [6] with pose refinement	0.850	30.5	0.658	27.7	0.887	30.1	0.818	27.1	0.702	26.6	0.801	27.5	0.827	28.9	0.853	29.5	0.799	28.5
<i>RadarSim</i> (ours)	<b>0.859</b>	<b>30.8</b>	<b>0.669</b>	27.5	<b>0.889</b>	<b>30.2</b>	<b>0.851</b>	<b>28.3</b>	<b>0.741</b>	<b>27.6</b>	<b>0.853</b>	<b>29.1</b>	<b>0.845</b>	<b>29.5</b>	<b>0.863</b>	<b>29.7</b>	<b>0.821</b>	<b>29.1</b>

Table 3. **Per-scene break down comparison with baselines RadarFields [2] and DART[6] with and without pose refinement.** *RadarSim* achieves significantly higher PSNR and SSIM on outdoor scenes (*Car*, *Statue*, *Wall*, *Outdoor A* *Outdoor B*) compared to baselines, as well as higher performance on indoor scenes (*Indoor C*, *Indoor D*), though the gains are less pronounced. This is because the effectiveness of leveraging surface normals to reproduce input radar recordings is reduced in indoor environments due to the increased presence of multipath reflections from corners and clutter. For Outdoor scene *Garden*, DART slightly outperforms *RadarSim* as this scene is dominated by inter-reflections between the metallic edge of the garden and ground as well as from bushes which diffusely reflects radar signal, making our normal-dependent model less effective.

## References

- [1] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. *CVPR*, 2022. 3
- [2] David Borts, Erich Liang, Tim Brodermann, 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 Conference Papers*, pages 1–10, 2024. 1, 4, 5
- [3] Fong et al. Panoptic nusenes: A large-scale benchmark for lidar panoptic segmentation and tracking. *arXiv*, 2021. 1
- [4] Rebut et al. Raw high-definition radar for multi-task learning. In *CVPR*, pages 17021–17030, 2022. 1
- [5] Xiao Fu, Wei Yin, Mu Hu, Kaixuan Wang, Yuexin Ma, Ping Tan, Shaojie Shen, Dahua Lin, and Xiaoxiao Long. Geowizard: Unleashing the diffusion priors for 3d geometry estimation from a single image. In *ECCV*, 2024. 4
- [6] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24118–24129, 2024. 1, 3, 4, 5
- [7] Andrew Kramer, Kyle Harlow, Christopher Williams, and Christoffer Heckman. Coloradar: The direct 3d millimeter wave radar dataset. *The International Journal of Robotics Research*, 41(4):351–360, 2022. 1, 2
- [8] Teck-Yian Lim, Spencer A. Markowitz, and Minh N. Do. Radical: A synchronized fmcw radar, depth, imu and rgb camera data dataset with low-level fmcw radar signals. *IEEE Journal of Selected Topics in Signal Processing*, 15(4):941–953, 2021. 1
- [9] Teck-Yian Lim, Spencer A Markowitz, and Minh N Do. Radical: A synchronized fmcw radar, depth, imu and rgb camera data dataset with low-level fmcw radar signals. *IEEE Journal of Selected Topics in Signal Processing*, 15(4):941–953, 2021. 2
- [10] Dong-Hee Paek, Seung-Hyun Kong, and Kevin Tirta Wijaya. K-radar: 4d radar object detection for autonomous driving in various weather conditions. *Advances in Neural Information Processing Systems*, 35:3819–3829, 2022. 2
- [11] Julien Rebut, Arthur Ouaknine, Waqas Malik, and Patrick Pérez. Raw high-definition radar for multi-task learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17021–17030, 2022. 2
- [12] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David McAllister, and Angjoo Kanazawa. Nerfstudio: A modular framework for neural radiance field development. In *ACM SIGGRAPH 2023 Conference Proceedings*, 2023. 4
- [13] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5481–5490. IEEE, 2022. 4
- [14] Ao Zhang, Farzan Erlik Nowruzi, and Robert Laganriere. Rad-det: Range-azimuth-doppler based radar object detection for dynamic road users. In *2021 18th Conference on Robots and Vision (CRV)*, pages 95–102. IEEE, 2021. 2