

# Supplementary Material: 3D Shape Reconstruction from Autonomous Driving Radars

Samah Hussein<sup>1</sup>  
samah.hussein@epfl.ch

Junfeng Guan<sup>1,3</sup>  
jguan1019@gmail.com

Swathi Shree Narashiman<sup>1</sup>  
swathi.narashiman@epfl.ch

Saurabh Gupta<sup>2</sup>  
saurabhg@illinois.edu

Haitham Hassanieh<sup>1</sup>  
haitham.alhassanieh@epfl.ch

<sup>1</sup> École Polytechnique Fédérale de  
Lausanne  
Switzerland

<sup>2</sup> University of Illinois Urbana-Champaign  
USA

<sup>3</sup> Bosch Research  
USA

## Contents

1	Background on Millimeter-Wave Radar Imaging	1
2	Loss Functions	3
3	Datasets	4
4	Experimental Hardware Setup	5
5	Additional Results	5
5.1	Impact of Number of Accumulated Frames	5
5.2	Failure Cases	6
5.3	SAR vs RFconstruct combination	6
5.4	Impact of Speed of Moving Cars	7
5.5	Additional Results from Randomly Selected Data Points	7
6	Limitations & Future Work	8

## 1 Background on Millimeter-Wave Radar Imaging

**3D Radar Imaging:** Millimeter-wave radars image the environment by transmitting radar waveforms to the 3D space and estimating reflected signal power in a 3D spherical coordinates defined by range ( $\rho$ ), azimuth angle ( $\phi$ ) and elevation angle ( $\theta$ ). The range and angular resolutions of the 3D radar heatmaps are determined by the signal bandwidth and the size of the antenna array, respectively.

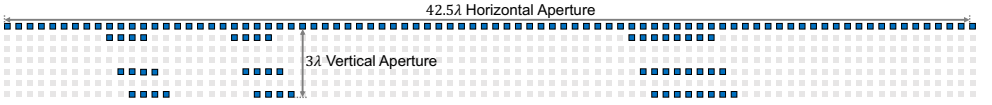


Figure 1: Virtual Antenna Array Topology of COTS mmWave Radar: Long 2D array ( $42.5\lambda \times 3\lambda$  aperture) with high azimuth resolution and low elevation resolution.

A 2D antenna array is needed to localize reflectors in the 3D spherical coordinates  $(r, \phi, \theta)$ , as the direction of reflector results in different phases at different antennas in the 2D array:

$$S_{m,k}(r) \propto e^{-j\frac{2\pi}{\lambda}(md\sin\theta\cos\phi + kd\cos\theta)} \quad (1)$$

where  $\lambda$  is the wavelength,  $d = \lambda/2$  is the separation between consecutive elements, and  $S_{m,k}(r)$  is the signal reflected by an object at distance  $r$  and then received on the antenna element index by  $(m, k)$  in the 2D array. Therefore, one can compute the reflected power  $x$  along the spherical angles  $\phi$  and  $\theta$  by adding a phase shift to the signal received on every antenna before combining the signals [10]:

$$x(r, \theta, \phi) = \sum_{m=0}^N \sum_{k=0}^N S_{m,k}(r) e^{j\frac{2\pi}{\lambda}[md\sin\theta\cos\phi + kd\cos\theta]} \quad (2)$$

**Resolution of Imaging Radars:** Although most mmWave radars today can achieve good depth resolutions (4-10 cm), their angular resolutions are nowhere near those of cameras and LiDARs. This is because unlike cameras or LiDARs, radars cannot directly distinguish lights coming from different angles using lenses or narrow laser beams. Instead, radars use an array of antennas that transmit and receive signals from all angles. Reflections from different angles are then resolved from the superposition by estimating the phase differences of the received signals across the antenna array, which is proportional to  $\cos\theta$ , where  $\theta$  is the incident angle.

However, angle estimation results in an ambiguity function as a *sinc* function. The width of the main-lobe of the *sinc* function is inversely proportional to the size of the antenna array, as is the angular resolution. The angular resolution of an antenna array with  $N$  elements separated by half wavelength ( $\frac{\lambda}{2}$ ) can be approximated by: *Angular Resolution*  $\approx \frac{2}{N\sin\theta}$ .

As a result, a point reflector in the scene is convolved with very wide *sinc* functions along both the azimuth and elevation axes in the resulting radar heatmaps, which eliminates all high-frequency shapes and details in the radar heatmaps. This is because, mathematically, convolution with a *sinc* is equivalent to multiplying the frequency domain with a rectangle function, effectively zeroing out high frequencies [10]. The side-lobes of the *sinc* function also create noise in the heatmap.

To achieve sufficient resolution in range, azimuth, and elevation, a mmWave radar requires a large 2D antenna array, which introduces significant hardware complexity. In practice, most COTS mmWave radars can form a large virtual antenna array along only one dimension, while the second dimension typically has a much smaller aperture and sparser antenna elements, often spaced beyond  $\lambda/2$ , as shown in figure 1. This leads to poor angular resolution and the presence of grating lobes, making accurate target detection along the second dimension highly challenging.



**Radar Point Cloud Processing:** The high-dimensional volumetric representation of 3D radar heatmaps are very sparse and noisy, yet comes with a very large volume, which puts a big burden on the data transfer and storage. Therefore, it is common practice in commercial mmWave radar devices to compress 3D radar heatmaps into a point-cloud format. The extraction of point clouds from heatmaps is done through a threshold process. For every voxel in the radar heatmap, if the corresponding reflection power is above a power threshold, the voxel is mapped to a point in the point cloud.

The challenge is to determine the power threshold, because a too high threshold leads to missed detections and loss of useful information, while a too low threshold results in very noisy point clouds that contain false alarms due to noise, side lobes of the 2D *sinc* function, and background clutter. The Constant False Alarm Rate (CFAR) algorithm and its variants such as Cell Averaging CFAR (CA-CFAR) and Ordered Statistic CFAR (OS-CFAR) are commonly used to convert radar heatmaps into point clouds. CFAR is an adaptive detection algorithm that adjusts the detection threshold in accordance with the measured background. Specifically, a CFAR detector estimates the noise floor for the cell under test by analyzing data from neighboring cells.

## 2 Loss Functions

The loss for our network is a combination of three weighted losses, classification loss, coarse loss, and fine loss. The classification loss is a softmax cross-entropy loss that measures the loss between the predicted class and the true class. For a batch of  $N$  examples, the classification loss is the average of the cross-entropy loss over the  $N$  examples:

$$L_{\text{class}} = -\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^K y_{ij} \log(\hat{y}_{ij}) \quad (3)$$

where  $y_{ij}$  is the true label and  $\hat{y}_{ij}$  is the predicted probability for class  $j$  of example  $i$ .

The coarse loss measures the loss between the output of the coarse stage in the decoder and the ground truth using the Earth Mover's Distance (EMD) metric [2]. EMD finds a bijection  $\phi : A \rightarrow B$  which minimizes the average distance between closest points. The optimal  $\phi$  is computationally expensive to find, which is why we implement an iterative approximation method to measure it. Moreover, we use EMD to compute the coarse loss the number of coarse points is smaller than the number of fine points, which accelerates the training process.

$$EMD(A, B) = \min_{\phi: A \rightarrow B} \frac{1}{|A|} \sum_{a \in A} \|a - \phi(a)\|^2 \quad (4)$$

Meanwhile, the fine loss is measured between the final output and the ground truth using Chamfer Distance (CD) [3] as the metric. CD measures the average bidirectional closest-point distance between the output and ground-truth point clouds. The Chamfer distance between two point sets  $A$  and  $B$  is given by:

$$CD(A, B) = \frac{1}{|A|} \sum_{a \in A} \min_{b \in B} \|a - b\|^2 + \frac{1}{|B|} \sum_{b \in B} \min_{a \in A} \|b - a\|^2 \quad (5)$$

Our combined loss function is a weighted sum of the three aforementioned components:

$$L = L_{\text{EMD}}(A_{\text{coarse}}, A_{\text{gt}}) + \alpha L_{\text{CD}}(A_{\text{fine}}, A_{\text{gt}}) + \beta L_{\text{class}}(C_{\text{pred}}, C_{\text{gt}})$$

### 3 Datasets

**(1) Radar Simulation.** We generate simulated data points by using a ray-tracing simulator provided in [9] and adapted for our radar and setup, namely two orthogonal 1-D virtual arrays. We capture 8 experiments per object in the dataset; in each experiment, the object is randomly positioned and oriented in the field of view. Moreover, we adjust the random orientations such that they are around the elevation axis, so that all of the experiments recover parts of objects that are typically observed from the perspective of a car, i.e., no flipped object. Finally, we use true-to-scale objects in the simulation set-up to ensure maximal compatibility with the real radar data. Each experiment consists of a number of frames, for each frame we simulate the horizontal and vertical heatmaps. Between frames, the radar antenna locations are changed in a linear trajectory parallel to the object to capture only one side of the object.

**(2) Geometric Perturbation Augmentation.** One drawback of simulation is that due to the complexity of the simulator design, the simulation time for these experiments can take a long time. This led to the need for additional data that is faster to generate. The simulator accurately resembles the noise patterns and ambiguities associated with radar data, such as sinc noise. However, it does not properly capture random noise generated from real scenarios and hardware limitations.

Thus, we use additional synthetic data generated from 3D models. We first randomly position a 'camera' around the 3D object, the camera should have a point of view similar to what a car would have as well. A snapshot is taken from this perspective, converted to a depth map, and sampled to generate the partial point cloud. This produces an ideal partial point cloud representation, which is repeated 8 times to generate 8 data points from different perspectives. We augment this representation to resemble radar data in two steps; the first step is to generate noise. As explained, mmWave radars have many artifacts in addition to their low angular resolution. We augment the partial point clouds to resemble these artifacts by replacing every point in the partial point cloud with multiple points randomly sampled from the surface of a sphere with a random, limited, radius that is centered around that point. This step creates noisier points that are slightly more dense, similar to our setup. The second step is to emulate specularity. As explained, specularity leads to the radar reflections missing some parts of objects, and while we partially overcome this challenge via temporal fusion, a practical trajectory will still suffer from specular signals since it cannot capture the object from every angle. To this end, we crop out multiple small spheres of the partial point cloud to emulate these parts as missing from the reflected signal. These spheres are randomly positioned so that at least one is near the center of the partial point cloud. This is based on the assumption that the center of most of our objects is the most specular part. The ground truth data are generated by uniformly sampling points from the 3D surface of the objects to represent the full shape.

**(3) Real-World Dataset.** We create a real-world radar dataset with 162 cars, 91 bikes, and 52 humans. We plan to release the full dataset in the format of processed heatmaps and 3D point clouds, as well as the corresponding positional information per frame. Our data collection experiments try to emulate a practical scenario as on self-driving cars by moving in mostly straight lines and observing objects only from one side, similar to what a typical car trajectory would be in reality. The trajectories of our partial point cloud experiments are short, straight segments with the radars facing in the general direction of an object of interest. The average trajectory length is about 3m, with a radar frame rate of approximately

60 fps. A single data point typically consists of 300~400 frames spanning about 6 seconds.

## 4 Experimental Hardware Setup

**Experimental Setup:** Our experimental setup shown in figure 4 in the paper consists of two TI MMWCAS radars [12], which contain 12 transmitters and 16 receivers. Moreover, we use a depth camera ZED 2i [13] that is deployed on the same moving platform as the two radars to enable careful tracking of experiments. The depth camera also contains an inertial measurement unit and positional tracking that we use to localize the radars at each captured frame. The two radars and camera are synchronized at the software level to trigger the capturing of the three sensors at the exact moment via a handshake. We also use a mobile phone to scan the full ground truth shapes of all test objects via the PolyCam [8], which utilizes the LiDAR and multiple cameras on the device. Our test experiments are done on static objects in different environments (indoor, outdoor) and different visibility conditions.

**Interference-Free Dual-Radar Operation.** To prevent interference between the two radar sensors, we carefully design the FMCW frame configurations so that the radars do not transmit simultaneously. This is achieved by alternating the transmission of chirps between the two radars within a single frame. Specifically, in a frame containing  $2N$  chirps, Radar 1 transmits and receives signals during the first  $N$  chirps and remains silent for the remaining  $N$  chirps. Conversely, Radar 2 remains silent during the first  $N$  chirps and transmits during the second  $N$  chirps. Experimental results confirm that any minor inconsistencies in chirp separation, caused by delays in the software handshake, are effectively mitigated by the low-pass filters integrated into the radar hardware. This ensures reliable operation without interference.

**Odometry.** There are several sensors that can be utilized for this purpose, including IMU, rotary encoder, or even camera-based Visual-Inertial odometry. Moreover, recent work employs the usage of mmWave radar heatmaps to extract accurate position information and perform SLAM [14]. To reduce complexity, we extract the positional tracking information from the co-located depth camera placed on the platform for ground truth imaging.

## 5 Additional Results

### 5.1 Impact of Number of Accumulated Frames

We study the effect of increasing the number of frames on the quality of the reconstructed shapes. Results indicate that longer trajectory times (number of frames) generally improve completion quality. However, the value gained by adding more frames decreases as the total number of frames increases, as can be shown in Figure 2. The plateau in the reconstruction improvement can be due to the practical, linear trajectories in which most of our experiments are conducted, where additional frames contribute less information about the object. As the trajectory progresses, new frames capture increasingly redundant information about the object's geometry. This redundancy arises because linear trajectories are limited to a specific viewpoint. Consequently, the marginal improvement in shape completion decreases with each additional frame.

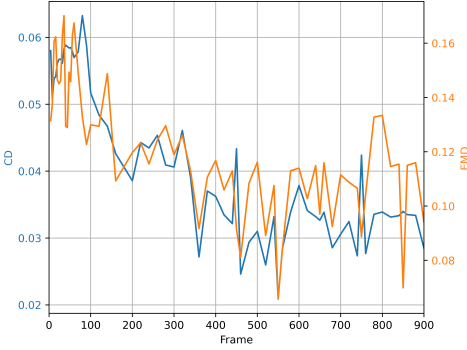


Figure 2: Reconstruction performance against input point cloud with increasing number of merged frames.

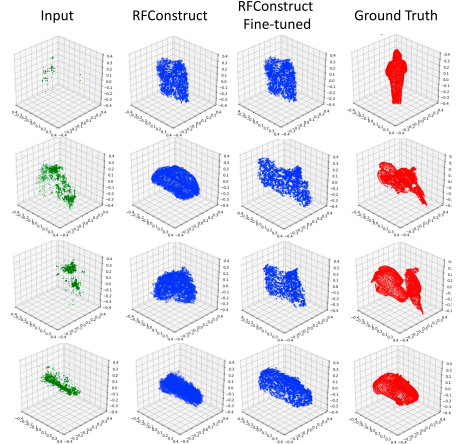


Figure 3: RFconstruct failure cases

## 5.2 Failure Cases

We were able to identify some specific conditions under which our system fails to provide correct reconstructions, as can be seen in figure 3. One case is when the input point cloud is exceedingly sparse, providing insufficient geometric information for accurate reconstruction. With too few points, the model struggles to infer meaningful structure, leading to incomplete or incorrect shapes. Failure can also occur as the Signal-to-Noise ratio of the input point cloud is very low, where noise points significantly outnumber valid ones. In this case, the model could mistakenly identify some of the noise points as valid points that represent the geometry of the object. This results in incorrect or distorted shapes. Addressing these limitations could involve incorporating mechanisms to handle sparse data more effectively and improve robustness to noisy inputs, such as advanced denoising techniques or prior knowledge of object structure.

Finally, the gap between the simulated training data and real data could lead to some misrepresentation of the object's geometry during reconstruction. This gap arises because the simulated training data often lacks the complexities and imperfections present in real-world scenarios, and are difficult to manufacture, such as measurement noise, environmental interference, or material properties that affect reflected signals. Which is why fine-tuned RFconstruct can provide a better reconstruction in some of the failure cases shown above.

## 5.3 SAR vs RFconstruct combination

As discussed in Section 1, Synthetic Aperture Radar (SAR) can combine radar information captured from different locations by utilizing scanning antennas. This approach improves resolution by emulating large antenna arrays using smaller ones. However, SAR is highly sensitive to small errors in antenna locations since half a wavelength at 77 GHz is less than 2 mm. Hence, we need millimeter level accuracy, otherwise the error in the phase of the signal leads to the accumulation of sinc noise rather than its cancellation.

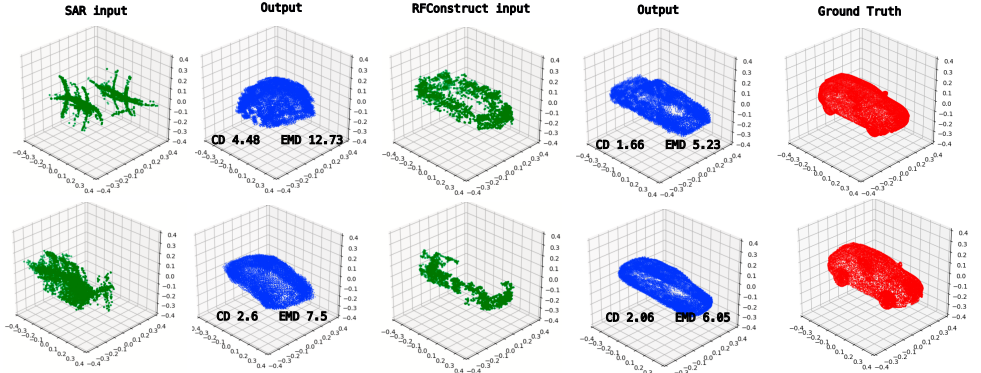


Figure 4: Comparing SAR point-clouds to RFconstruct point-clouds and completion performance.

We conduct experiments with SAR imaging using our experimental setup. As shown in Fig 4, the coherent combination of the collected frames amplifies sinc noise and results in prominent grating lobes. In contrast, our proposed combination method produces accurate, though sparse, point clouds. Furthermore, when both point clouds—from the coherent combination and our RFconstruct method—are used as inputs to complete the 3D shape, the output derived from the RFconstruct point cloud is both quantitatively and qualitatively better than that obtained from the coherent combination. In particular, the CD can decrease from 4.48 to 1.66 and the EMD from 12.73 to 5.23.

## 5.4 Impact of Speed of Moving Cars

To evaluate RFconstruct’s performance in the case of non-static vehicles, we run an experiment where we vary the speeds of the ego vehicle and the vehicle we are trying to image. The vehicles move in the opposite direction at the same constant speed, which we vary to 10 km/h, 20 km/h, and 30 km/h, resulting in relative speeds of 20 km/h, 40 km/h, and 60 km/h, respectively. We use a sliding window to accumulate radar frames. Figure 5 shows the reconstructed car shape at the above speeds when the car is fully within the field of view of the radar and when the car is at the edge of the FOV of the radar. While in both cases the reconstruction quality degrades as the speed increases, when the car is inside the FoV, the radar can still capture enough frames to achieve reasonable reconstruction. However, when the car is at the edge of the FoV (entering the FoV), the quality degrades quickly with speed to become unrecognizable at 60 km/h. Note that in the case where the target vehicle is static, a good reconstruction is achieved in both cases, as RFconstruct was trained on static scenarios. To address this, one must estimate the relative speed using Doppler and dynamically adjust the sliding window size and frame sampling rate. One must also retrain RFconstruct for moving targets. We, however, leave this for future work.

## 5.5 Additional Results from Randomly Selected Data Points

### 5.5.1 RFconstruct with Bounding-Box Priors

In figure 6 we show results for RFconstruct and fine-tuned RFconstruct for randomly selected data points from the testing set. We show both the coarse and fine outputs from the network.

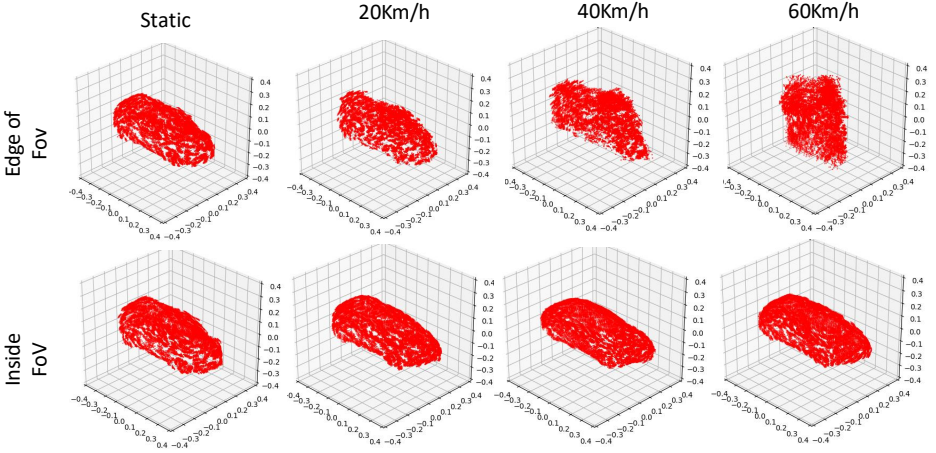


Figure 5: Qualitative results for cars in motion.

While the basic system without fine-tuning produces representative reconstructions for most cases, Fine-tuning improves the reconstructions and brings the representations much closer to the actual ground truth shapes.

### 5.5.2 RFconstruct without Bounding-Box Priors

In the case of testing without bounding box priors, the reconstruction is more challenging as more ambiguity arises about the object type, orientation and dimensions. In figure 7, we show random samples of the data tested with RFconstruct and fine-tuned RFconstruct trained without bounding box priors.

## 6 Limitations & Future Work

While RFconstruct takes major steps towards enabling 3D shape reconstruction from mmWave radars in autonomous driving, it still has several limitations that require addressing before it can be fully realized in practice.

- *Moving targets:* Our current implementation of RFconstruct is trained for static objects. While our demo and results on different car speeds show promising results by using a sliding window to accumulate radar frames, the quality of the reconstruction does degrade with the speed of the cars. Imaging moving objects requires more research as we cannot easily fuse temporal radar frames. We will need to develop algorithms to estimate and compensate for the relative motion and relative positions of the objects with respect to the ego car. We will also explore the tradeoff between smearing the object and combating specularity in setting the temporal fusion window, which depends on the relative speed and relative direction of motion.
- *Generalizing to other shapes & environments:* We only trained and tested RFconstruct for three classes of objects that represent the most common moving objects on the road that present safety risks. To make RFconstruct more general, we need to extend it to



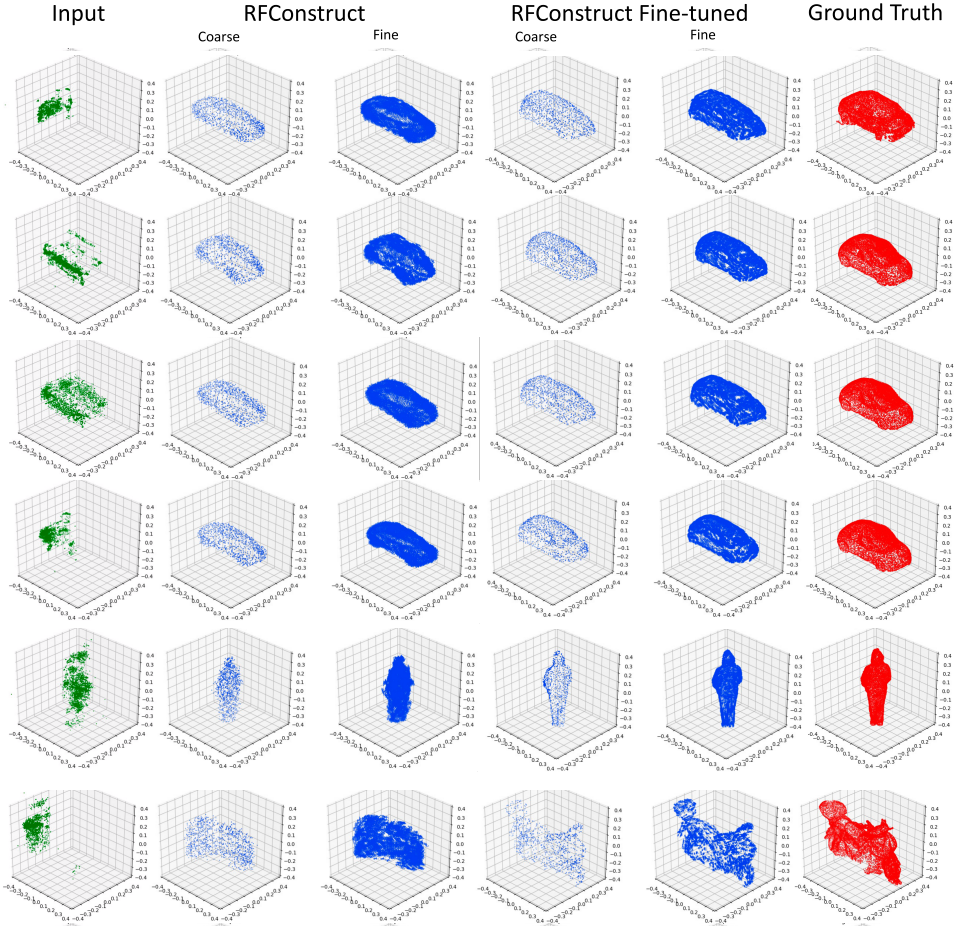


Figure 6: Qualitative results for RFconstruct and RFconstruct fine-tuned on real radar data for randomly selected data points.

new classes like trees, road signs, fire hydrants, trash cans, large trucks, etc. We also need to update the neural network architecture and training to generalize to different streets and roads with different building structures.

- *Incorporating Doppler:* RFconstruct currently does not leverage the Doppler information from radar. Incorporating Doppler would allow us to separate nearby objects along the Doppler domain (cars, bikes, and pedestrians move at different speeds). Doppler also allows us to estimate the speed of various objects in the scene, which is essential for fusing temporal radar frames of moving objects.
- *Pedestrian posture:* While RFconstruct can recover the orientation of the cars and bikes in the scene, it cannot estimate the orientation or posture of pedestrians. Currently, it only estimates a rough silhouette but cannot accurately recover the human posture, which can be useful for driving cars to estimate the direction the pedestrian is walking. We can build on the extensive literature on human pose estimation [10, 11, 12, 13, 14, 15] from radar signals and incorporate it into RFconstruct.

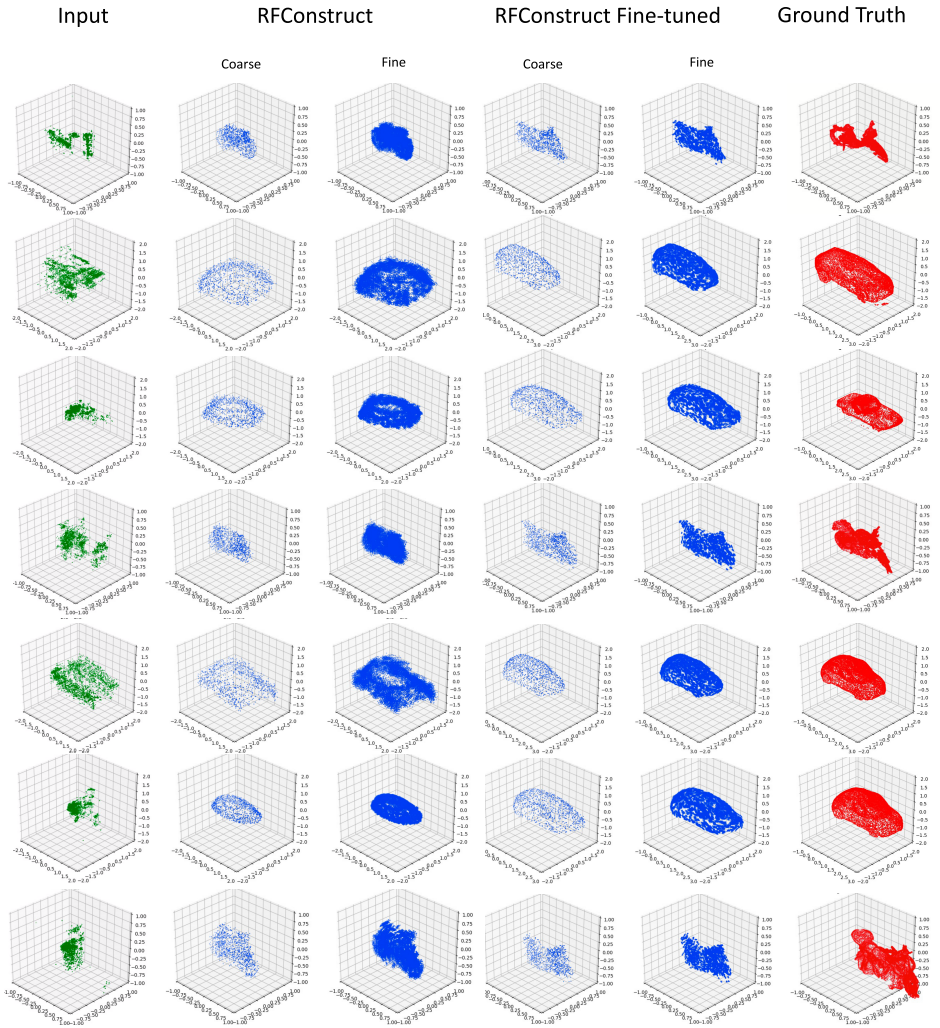


Figure 7: Qualitative results for RFConstruct and RFConstruct fine-tuned on real radar data **without** bounding-box priors for randomly selected data points.

## References

- [1] Anjun Chen, Xiangyu Wang, Shaohao Zhu, Yanxu Li, Jiming Chen, and Qi Ye. mm-body benchmark: 3d body reconstruction dataset and analysis for millimeter wave radar. In *Proceedings of the 30th ACM International Conference on Multimedia*, MM '22, page 3501–3510, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450392037. doi: 10.1145/3503161.3548262. URL <https://doi.org/10.1145/3503161.3548262>.
- [2] Haoqiang Fan, Hao Su, and Leonidas Guibas. A point set generation network for 3d object reconstruction from a single image. In *2017 IEEE Conference on Computer*



- Vision and Pattern Recognition (CVPR)*, pages 2463–2471, 2017. doi: 10.1109/CVPR.2017.264.
- [3] Junfeng Guan, Sohrab Madani, Suraj Jog, Saurabh Gupta, and Haitham Hassanieh. Through fog high-resolution imaging using millimeter wave radar. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11461–11470, 2020. doi: 10.1109/CVPR42600.2020.01148.
  - [4] Ziyang Hong, Yvan Petillot, and Sen Wang. Radarslam: Radar based large-scale slam in all weathers. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5164–5170, 2020. doi: 10.1109/IROS45743.2020.9341287.
  - [5] Hao Kong, Xiangyu Xu, Jiadi Yu, Qilin Chen, Chenguang Ma, Yingying Chen, Yi-Chao Chen, and Linghe Kong. m3track: mmwave-based multi-user 3d posture tracking. In *Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services, MobiSys '22*, page 491–503, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450391856. doi: 10.1145/3498361.3538926. URL <https://doi.org/10.1145/3498361.3538926>.
  - [6] Hamid Krim and Mats Viberg. Two Decades of Array Signal Processing Research. *IEEE Signal Processing Magazine*, 1996.
  - [7] Shih-Po Lee, Niraj Prakash Kini, Wen-Hsiao Peng, Ching-Wen Ma, and Jenq-Neng Hwang. Hupr: A benchmark for human pose estimation using millimeter wave radar. In *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 5704–5713, 2023. doi: 10.1109/WACV56688.2023.00567.
  - [8] Polycam Inc. Polycam 3D Scanner, LiDAR, 360. <https://learn.poly.cam/about>, 2024. [Online; accessed jun-24-2024].
  - [9] Xie Qian, Deng Qianyi, Cheng Ta-Ying, Zhao Peijun, Patel Amir, Trigoni Niki, and Markham Andrew. mmpoint: Dense human point cloud generation from mmwave. In *The British Machine Vision Conference (BMVC)*, 2023.
  - [10] Akbar Sayeed and Nader Behdad. Continuous aperture phased mimo: Basic theory and applications. In *2010 48th annual allerton conference on communication, control, and computing (Allerton)*, pages 1196–1203. IEEE, 2010.
  - [11] Stereolabs Inc. ZED 2. <https://www.stereolabs.com/products/zed-2>, 2022. [Online; accessed jun-20-2024].
  - [12] Texas Instruments Inc. MMWCAS-RF-EVM mmwave cascade imaging radar rf evaluation module. <https://www.ti.com/tool/MMWCAS-RF-EVM>, 2024. [Online; accessed jun-24-2024].
  - [13] Hongfei Xue, Yan Ju, Chenglin Miao, Yijiang Wang, Shiyang Wang, Aidong Zhang, and Lu Su. mmmesh: towards 3d real-time dynamic human mesh construction using millimeter-wave. In *Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services, MobiSys '21*, page 269–282, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450384438. doi: 10.1145/3458864.3467679. URL <https://doi.org/10.1145/3458864.3467679>.

- 
- [14] Mingmin Zhao, Tianhong Li, Mohammad Abu Alsheikh, Yonglong Tian, Hang Zhao, Antonio Torralba, and Dina Katabi. Through-wall human pose estimation using radio signals. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7356–7365, 2018. doi: 10.1109/CVPR.2018.00768.
- [15] Mingmin Zhao, Yonglong Tian, Hang Zhao, Mohammad Abu Alsheikh, Tianhong Li, Rumen Hristov, Zachary Kabelac, Dina Katabi, and Antonio Torralba. Rf-based 3d skeletons. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication, SIGCOMM '18*, page 267–281, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450355674. doi: 10.1145/3230543.3230579. URL <https://doi.org/10.1145/3230543.3230579>.