High Resolution Point Clouds from mmWave Radar

Akash Prabhakara*, Tao Jin*, Arnav Das†, Gantavya Bhatt†, Lilly Kumari†,
Elahe Soltanaghai‡, Jeff Bilmes†, Swarun Kumar* and Anthony Rowe*§

* Carnegie Mellon University  † University of Washington
‡ University of Illinois, Urbana-Champaign  § Bosch Research

Abstract—This paper explores a machine learning approach on data from a single-chip mmWave radar for generating high resolution point clouds—a key sensing primitive for robotic applications such as mapping, odometry and localization. Unlike lidar and vision-based systems, mmWave radar can operate in harsh environments and see through occlusions like smoke, fog, and dust. Unfortunately, current mmWave processing techniques offer poor spatial resolution compared to lidar point clouds. This paper presents RadarHD, an end-to-end neural network that constructs lidar-like point clouds from low resolution radar input. Enhancing radar images is challenging due to the presence of specular and spurious reflections. Radar data also doesn’t map well to traditional image processing techniques due to the signal’s sinc-like spreading pattern. We overcome these challenges by training RadarHD on a large volume of raw I/Q radar data paired with lidar point clouds across diverse indoor settings. Our experiments show the ability to generate rich point clouds even in scenes unobserved during training and in the presence of heavy smoke occlusion. Further, RadarHD’s point clouds are high-quality enough to work with existing lidar odometry and mapping workflows.

I. INTRODUCTION

Lidar is often considered the gold standard in terms of sensors used for mapping, localization, and collision avoidance in robotics. A key enabler is its ability to generate low-noise and high-density point clouds, which can be easily tracked from one position to the next. Despite its ubiquitous use, lidar, just like visible light cameras, fail when used in environments with occlusions—e.g. when robots navigate thick fog, smoke or dust, say for search-and-rescue, disaster recovery and firefighting. Applications demand sensors that see past occlusions and sense the world in high fidelity. RadarHD is inspired by important recent work on using neural networks on mmWave radar for individual applications: odometry and mapping [10], [11]. However, unlike these systems that target specific higher-level applications, RadarHD targets a broader and more general problem: generating high resolution point cloud data directly from radar I/Q streams that is as good as what a lidar would output (albeit working in lidar-denied scenes). Our approach has two key benefits over per-application end-to-end learning: (1) Point clouds provide an interpretable, easy to understand output. For instance, it’s more intuitive to debug and reason about point cloud errors rather than odometry errors, especially when both are output of a machine learning pipeline. (2) High quality point clouds enable a more general representation that can replace lidar in existing point cloud processing pipelines for several tasks beyond just odometry and mapping such as object detection and classification.

In designing RadarHD, we encountered two challenges. First, raw radar measurements are impacted by various spurious artifacts—sidelobes from strong reflectors that create sinc-like patterns across azimuth due to its poor azimuth
resolution (see Fig. 1), specular reflections from certain objects, and other processing artifacts [12]. Eliminating these artifacts to recover the true objects is crucial for constructing a dense, accurate point cloud. Second, radar images are coarse – meaning that they struggle with resolving sharp environmental features with high angular resolution. In other words, the data from a low resolution radar is quite different from low resolution camera images, where naïve super resolution such as interpolation would give a sensible result.

RadarHD overcomes these challenges by posing a supervised learning problem where large datasets of radar-lidar pairs collected on identical scenes, are used to inform radar to distinguish true objects from noise/artifacts. RadarHD’s core contribution is the customization of the entire neural network pipeline – input/output representation, architecture and loss functions – to tackle each of the challenges mentioned above.

To implement RadarHD, we collect a large repository (200k pairs) of raw I/Q radar data from TI AWR1843 mmWave radars paired with lidar point clouds across different indoor and outdoor environments for generalization. RadarHD’s evaluation reveals low point-cloud error (24 cm) versus lidar ground-truth and 3.5× superior to traditional radar point-clouds. We also demonstrate the quality of our point clouds with two applications: odometry and mapping, using Google Cartographer [13].

Contributions: RadarHD makes three key contributions:

- Application of a super resolution model for generating lidar-like point clouds from low resolution radar.
- A detailed evaluation of the system in new, unseen environments and severe occlusion such as smoke.
- A large repository of raw radar I/Q and lidar pairs along with source code.

II. RELATED WORK

Radar Super Resolution: The mm-level wavelength and the wide bandwidth available at mmWave frequency range provide high ranging accuracy and sensitivity. Combined with the robustness of mmWave radars to different lighting and weather conditions, mmWave radar is a popular option for sensing purposes [14], [15], [16], [17], [18], [19], [20], [21], [22]. In radar, high resolution is usually achieved by using Synthetic Aperture Radar (SAR) [23], [24], [3], [4], [5] or sensor fusion such as integrating radar and camera/lidar [25], [26], [27], [28]. While SAR is used in mobile contexts such as satellite imagery and automotive [29], inaccurate motion information causes errors in the synthesized image [30]. For more portable applications (e.g: future small robots, light-weight UAVs etc.) that we envision, mm-accurate (on the scale of mmWave wavelength) motion information can be expensive to obtain. Moreover, our applications need high resolution images even when the radar is not being moved/temporarily static. More recently, techniques leverage deep learning [6], [31], [32], [33], [34] to perform radar super resolution. [34] uses deep learning to only keep robust points from the range-doppler spectrum. We instead tackle super-resolution and seek to create lidar-like point cloud which not only have true radar points but other synthetically generated points that boost the resolution. HawkEye [6] is the closest related work to RadarHD. However, it relies on input data obtained from mechanically scanning the mmWave radar on a large aperture slider to perform SAR [6], [35]. Other works such as [31], [33] just like [6] deal with static radar platform and single object setup. That is, a static radar is looking at a single object (static car/person) whose 3D point cloud is of interest. In such setups where radar platform and object are static, a ground truth can be obtained from a SAR system [31], and low resolution radar could be trained to generate SAR-like output. In contrast, we want to generate high resolution output just from a single-chip radar when (1) radar is static/arbitrarily moving and (2) in real, complicated environments. This calls for rethinking how we collect data, what our ground truth is and design the entire learning pipeline to deal with radar artifacts that show up due to real, complicated environments.

Radar Odometry and Mapping: State-of-the-art radar odometry techniques [36], [37], [38], [39], [40], [41], [42], [43], [44], [45], [46] mainly rely on scan matching, in which spatial features from the radar images are matched against previous scans or a pre-determined map. However, due to spurious reflections and artifacts in radar data, the accuracy of these methods significantly drop for low resolution radars. Follow-up research in this area relies on more complex modeling such as using velocity estimates from radar scans [47], [48], [49], [50], [51], [52], the fusion of radar and IMU data based on Extended Kalman Filtering [53], [54], [41], [55], or fusion of radar and RGB camera [56], [57], [58] to overcome radar limitations. Few other approaches involve end-to-end deep learning approaches for fusing Constant False Alarm Rate (CFAR) radar point clouds and IMU to obtain either odometry [10] or mapping [11], [59] individually. RadarHD, on the other hand, seeks to solve an orthogonal problem: replacing CFAR points by creating higher resolution point clouds. These point clouds are general-purpose and can be fed into multiple lidar processing pipelines such as odometry, mapping, object detection, classification etc.

Radar-Lidar Datasets: Most publicly available radar and lidar datasets capture outdoor environments when mounted on a car [60], [61], [62]. Some of them use non-compact, bulkier mechanically scanning radar [60]. We want a large repository of single-chip radar and lidar pairs in real, complicated environments for robotic applications. While there have been datasets for radar odometry [10], mapping [11] and activity recognition [17], ground truth lidar point cloud is not commonly exposed. More recently, [63], [64] provide single chip radar datasets with monocular and stereo camera largely for automotive purposes. [65] is the closest that comes to the raw radar-lidar dataset that we desire. However, the indoor data samples (~15k radar-lidar pairs) are only a fraction of the entire dataset. To allow for larger volume of training data, custom design test cases and test robustly, we collect our own dataset.
A. Radar Data Representation

We first consider the problem of representing radar signals as input to the ML architecture. To do so, we briefly describe our radar platform’s capabilities and typical sensor output.

Radar Platform: Our mmWave radar [67] provides raw I/Q data streams that can undergo further processing as needed. A typical radar processing pipeline [68] involves a spatial Fourier transform that outputs a 2D heatmap with intensity of the reflected radar signal across range and azimuth. Traditionally, the heatmap gets thresholded to a “point cloud”. Constant False Alarm Rate (CFAR) is one such threshold detector that keeps robust objects and eliminate noisy ones. Fig. 2 shows CFAR thresholding applied to radar heatmap.

Choice of Input Representation: At this point, we have a choice – do we feed in raw I/Q inputs directly or should we send processed point cloud data. On one hand, inputting raw I/Q requires the model to understand and learn phase information and to learn some fairly obvious initial steps (e.g. a Fourier Transform). On the other hand, sending in a highly thresholded heatmap may filter out important information that values below the threshold carry, e.g. feeble objects masked by sinc lobes from stronger reflections.

RadarHD instead takes an approach in between these two extremes. Specifically, RadarHD applies a very low threshold to the processed heatmap so that it preserves dominant reflectors, feeble ones and many artifacts. Our objective is to retain a significant portion of the heatmap, including feeble reflectors while leaving it to the ML model to learn and filter out spurious artifacts. RadarHD chooses a threshold such that extremely weak points are omitted, but is still low enough to propagate radar’s artifacts, strong and feeble reflectors. For context in Fig. 2 CFAR thresholding has 110 non-zero pixels and RadarHD’s thresholding has 1606 non-zero pixels.

Polar vs Cartesian Representation: We note that the thresholded image above is actually in polar format i.e. range-azimuth, but is shown in Fig. 2 in Cartesian format for easy understanding. Given that the output we desire for a point cloud is Cartesian, one may consider the Cartesian representation as the natural choice of representation.

However, radar inherently measures radially and side lobes arising from a strong reflector also spread azimuthally. To capture these radial and azimuthal variations, one would need radial/azimuthal processing. But the primary learning element in convolutional layers in machine learning is a filter that performs 2D correlation across the height and width of the input. To leverage this to our advantage, we choose a polar format so that the filters then naturally traverse along the range and azimuth when they go across height and width, respectively. We thus have thresholded points on the heatmap with range, azimuth, and intensity arranged into an image with range (0-10 m) along rows and azimuth (-90° to 90°) along columns (Fig. 3). The radar images (64 × 256) are narrower than lidar (512×256) because of the poorer azimuth resolution.
Ground Truth

Fig. 4. Qualitative Ablation Study: Dice Loss helps in sharpening lines, but Naïvely one could use single frame inference and classical filtering to ensure that objects do not appear and disappear. We perform this filtering through the network by incorporating past radar frames (that offer multiple viewpoints if radar is moving) as input while performing super resolution on the current frame. Our design is to exploit the variable number of input image channels and stack the past frames as input channels to allow for understanding of each radar frame and modeling persistence across frames. We empirically analyze and find 40 past frames (2 seconds history) to be sufficient for enforcing persistence. Even in static cases, using past frames lead to smooth and less jittery output.

C. Neural network training methodology

We carefully select loss functions to preserve features such as sharp lines that appear in lidar images (thin lines of white pixels against black background). We use a combination of various loss functions for achieving different objectives.

Pixel wise loss: To compare two images, one ground-truth label and one output from the network, we first consider the most standard loss function – pixel wise loss. Our ground truth labels are binary lidar images. We compare this binary image against the final sigmoid layer output from the network. We use mean Binary Cross-Entropy (BCE) over all pixels. The objective of this pixel wise loss is to force each pixel to match the expected output. Fig. 4 shows that BCE alone generates an acceptable output, but the lines and boundaries are not as sharp as the ground truth.

Dice loss: To promote crisp and sharp lines in the output image, we draw from Dice loss [70] used in computer vision tasks like boundary detection [71]. For each pixel in ground-truth $g_i$ and network output $o_i$, Dice loss for $N$ pixels is:

$$D = 1 - \frac{2 \sum_{i=1}^{N} o_i g_i}{\sum_{i=1}^{N} o_i^2 + \sum_{i=1}^{N} g_i^2}$$

Here, the numerator finds the loss pixel wise and is maximized when both $o$ and $g$ are identical. The denominator keeps a global view of total number of points that are 1. The loss promotes maximizing the intersection between $o$ and $g$ and penalizes the union of $o$ and $g$. This forces the network to output 1 exactly where the ground truth is 1, while remaining 0 where the ground truth is 0. This enables a sharper and crisper prediction than pixel wise cross entropy. We analyze and find trade-off between BCE and Dice loss (Fig. 4). More Dice loss leads to eliminating certain important features.
IV. IMPLEMENTATION

System Hardware: RadarHD was implemented using TI mmWave radar AWR1843, a state-of-the-art single-chip radar with a theoretical range resolution of 3.75 cm and azimuth resolution of 15°. RadarHD’s objective is to improve this azimuth resolution. We use the AWR1843 together with DCA1000EVM to collect raw I/Q samples.

Testbed: Our testbed consists of radar, lidar for ground truth and camera for debugging - all time-synced. Our testbed is mounted on a mobile testing rig. Our entire data repository consists of about 200k radar I/Q - lidar pairs collected across a total area of 5147 m² which we believe will be extremely useful to the research community.

Ground Truth: We use Ouster OS 0 - 64 beam mechanical scanning lidar for our ground truth. The lidar is configured to work at 0.35° azimuth resolution. We only use the forward-facing lidar points for super resolution and we also restrict the lidar’s elevation FoV to be within +/-30cm.

Baselines: We use Constant False Alarm Rate (CFAR) based thresholding with different thresholds as our baselines. Today, CFAR is widely used for collecting radar point clouds. CFAR is an ideal baseline as: it is not machine learning based, relies purely on radar signals (no IMU) and works when radar is either static/mobile (unlike SAR). We specifically implement Cell-Averaging CFAR [72].

Train and Test Data: We train our system in a large office space that includes furniture, electronics, walls, conference rooms, and cubicles, with a total area of 839 m². All of our tests are performed on unseen data in (1) same environment with different trajectories collected across different days, (2) similar environment in different office spaces with different cubicle structures, and (3) different environment, which is totally new and includes building lobbies and outdoor environments. Across all our data we use 28 trajectories for training (561 meters long with 22784 image pairs in total) and 39 diverse trajectories for testing (714 meters long with 36779 image pairs in total) to allow for testing robustly.

V. RESULTS

A. Point cloud Comparison

Method: After training on rich office space environment, we run RadarHD on all the diverse test samples. To compare against lidar point cloud, we first convert the threshold the range-azimuth output image to obtain a list of points with their (x,y) location. We then compare point cloud error using two popularly used point cloud similarity metrics [73]: (1) Chamfer distance [74]: finds the nearest neighbor for each point in one point cloud to the other, and takes the mean of all these distances to get an error for each point cloud pair. (2) Modified Hausdorff Distance [75]: which also finds the nearest neighbors and obtains the median neighbor distance.

Comparison to baseline: Here, we show our performance in the floor-wide office environment on 19 different trajectories against different CFAR thresholds. This includes 18k radarradar point cloud pairs, each over a 10x20 meter area.

As seen in Fig.5, we obtain a 0.24 m modified-Hausdorff median error and 0.36 m Chamfer median error. CFAR, on the other hand, varies depending on the threshold. A low threshold like 1dB threshold creates point cloud 5x denser than lidar, while a high threshold like 8dB would just have 10% of the number of points captured using lidar. Despite varying density levels across these extremes, none of them have any structural similarity to the ground truth lidar point cloud. So from 1dB to 8dB, as the threshold increases and density decreases, both point cloud error metric Cumulative
To study the impact of occlusion, we build a smoke chamber around the testing rig and drop smoke pellets in it to create dense smoke. The radar is kept static and same scene is captured without smoke and varying levels of smoke. We use onboard camera to judge the intensity of smoke. We notice that even with 1 smoke pellet, that generates 500 cubic feet/min, lidar does not receive any points. However, the collected radar signals in smoke are almost identical to that without smoke even for the densest smoke we could create using 4 pellets. As the collected radar signal remains the same, we expect similar performance as without smoke.

Fig. 7 validates this by showing that RadarHD’s performance doesn’t degrade, up to densities we could create.

B. Odometry and Mapping Comparison

**Method:** Using the high quality point clouds generated by RadarHD, we next show 2 downstream tasks that RadarHD will enable in scenarios where lidars fail: odometry and mapping. Since our points are lidar-like, we evaluate this by feeding our point clouds, without adding any other sensor (e.g IMU), into existing lidar SLAM frameworks such as Google Cartographer [13]. We obtain the 3-DoF pose in 2D (translation (x,y) and rotation) and map from Cartographer.

**Odometry:** We evaluate odometry against lidar and benchmark CFAR point clouds using Absolute Trajectory Error - ATE (see Fig. 10). In all cases, including different environments, the odometry accuracy of RadarHD outperforms that of CFAR regardless of threshold. Qualitatively, one can clearly see the difference between RadarHD odometry and CFAR odometry in Fig. 9. RadarHD achieves performance comparable to 0.8 m reported in radar+IMU pipelines dedicated for odometry in past work [10].

**Mapping:** We benchmark the mapping performance by identifying keypoints that point to the same physical feature in the real world, such as corners of a room, and then calculate the Euclidean distance error of corresponding keypoints between RadarHD and ground truth. Fig. 9 shows a qualitative comparison of a map generated from one trajectory. It is clear that CFAR does not provide any meaningful features to extract keypoints while RadarHD achieves a structurally similar map compared to lidar. Fig. 10 shows the Euclidean distance error between keypoints across different environments. Good performance on odometry/mapping is possible only because of artifact-free, meaningful point clouds generated by RadarHD. RadarHD also allows for visual debugging of point clouds in case of poor odometry/mapping performance.

## VI. Conclusion and Future Work

RadarHD creates a lidar-like high resolution point cloud from low resolution single-chip mmWave radar input for use in robotic applications where lidar fails. RadarHD designs a machine learning pipeline for this task and overcomes the challenges arising from radar artifacts by choosing design parameters. We show our rich point cloud in a variety of scenes - completely new environments and in occlusions such as smoke. We collect a large dataset of radar-lidar raw data pairs, which is useful for other perception tasks. In the future, we hope to solve other challenges in enabling RadarHD to be an invaluable asset in situations where lidar fails. This includes moving beyond 2D and generating 3D point clouds, tackling highly dynamic scenes, and dealing with 3-dimensional mobile platforms (e.g. UAVs).

**Acknowledgements:** This research was supported by NSF (2030154, 2106921, 2007786, 1942902, 2111751), Bosch, CONIX Center, DARPA-TRIAD, ARL and CMU-MFI.

### Table: Odometry and Mapping Comparison

<table>
<thead>
<tr>
<th>ATE (meter)</th>
<th>CFAR</th>
<th>RadarHD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean 5.02</td>
<td>3dB</td>
<td>Same env.</td>
</tr>
<tr>
<td></td>
<td>6dB</td>
<td>Similar env.</td>
</tr>
<tr>
<td></td>
<td>Diff env.</td>
<td>1.30</td>
</tr>
<tr>
<td></td>
<td>Diff env.</td>
<td>1.28</td>
</tr>
<tr>
<td>RMSE 5.74</td>
<td>Same env.</td>
<td>3.98</td>
</tr>
<tr>
<td></td>
<td>Similar env.</td>
<td>1.35</td>
</tr>
<tr>
<td></td>
<td>Diff env.</td>
<td>1.52</td>
</tr>
<tr>
<td></td>
<td>Diff env.</td>
<td>1.44</td>
</tr>
<tr>
<td>Median 5.70</td>
<td>Same env.</td>
<td>2.42</td>
</tr>
<tr>
<td></td>
<td>Similar env.</td>
<td>0.80</td>
</tr>
<tr>
<td></td>
<td>Diff env.</td>
<td>1.22</td>
</tr>
<tr>
<td></td>
<td>Diff env.</td>
<td>1.21</td>
</tr>
<tr>
<td>Std 2.79</td>
<td>Same env.</td>
<td>2.35</td>
</tr>
<tr>
<td></td>
<td>Similar env.</td>
<td>0.87</td>
</tr>
<tr>
<td></td>
<td>Diff env.</td>
<td>0.80</td>
</tr>
<tr>
<td></td>
<td>Diff env.</td>
<td>0.65</td>
</tr>
</tbody>
</table>

![Fig. 10. Odometry (Absolute Trajectory Error - ATE) and Mapping quantitative results.](image-url)

Distribution Functions (CDFs) shift to the left. However, because the highest threshold generates a very sparse point cloud and does not resemble ground truth, the CDFs do not shift any more to the left with an increase in the threshold. In fact, beyond 8dB, the number of points was less than 1% that of lidar. Our generated point cloud not only improves upon CFAR by 3.5× (mod-Hausdorff) and 2.7× (Chamfer), but is also structurally more similar to the ground truth as seen in Fig. 8 unlike Fig. 7.

**Generalization:** We compare the performance of point cloud generation along these 19 diverse trajectories against those obtained in similar and different environments having trained only on same environments. We use 7.5k point cloud pairs, each for similar and different environments.

Fig. 6 shows the change in performance in new environments. We see a median error of 0.75 m (mod-Hausdorff) and 0.8 m (Chamfer) for similar environments and 0.94 m (mod-Hausdorff) and 1.03 m (Chamfer) for different environments. This tells us that these point clouds generated are not quite as accurate as when tested in same environment. At first glance, we observe that these medians are similar to the medians for some CFAR CDFs in Fig. 5. However, we would like to point out three important reasons why RadarHD’s point clouds are still superior. First, the CFAR CDFs start on the x-axis at 0.4 m; in contrast, even for similar and different environments, we see that the CDFs start at 0.08 m. This shows that there is a significant fraction of the point clouds that is accurately inferred by our system. Second, because both Chamfer distance and modified Hausdorff distance have a nearest neighbor point association, structural similarity is not entirely captured [76]. However, we can qualitatively see from Fig. 8 that our system indeed generates meaningful points. Third, to quantitatively show the impact of improved accuracy, we compare RadarHD against CFAR in two key applications - odometry and mapping in Sec. V-B.

**Smoke:** To study the impact of occlusion, we build a smoke chamber around the testing rig and drop smoke pellets in it to create dense smoke. The radar is kept static and same scene is captured without smoke and varying levels of smoke. We use onboard camera to judge the intensity of smoke.

We collect a large dataset of radar-lidar raw data pairs, which is useful for other perception tasks. In the future, we hope to solve other challenges in enabling RadarHD to be an invaluable asset in situations where lidar fails. This includes moving beyond 2D and generating 3D point clouds, tackling highly dynamic scenes, and dealing with 3-dimensional mobile platforms (e.g. UAVs).
REFERENCES


