RadCloud: Real-Time High-Resolution Point Cloud Generation Using Low-Cost Radars for Aerial and Ground Vehicles

David Hunt, Shaocheng Luo, Amir Khazraei, Xiao Zhang, Spencer Hallyburton, Tingjun Chen, Miroslav Pajic

Abstract— **In this work, we present RadCloud, a novel *real-time* framework for directly obtaining higher-resolution lidar-like 2D point clouds from low-resolution radar frames on resource-constrained platforms commonly used in unmanned aerial and ground vehicles (UAVs and UGVs, respectively); such point clouds can then be used for accurate environmental mapping, navigating unknown environments, and other robotics tasks. While high-resolution sensing using radar data has been previously reported, existing methods cannot be used on most UAVs, which have limited computational power and energy; thus, existing demonstrations focus on offline radar processing. RadCloud overcomes these challenges by using a radar configuration with 1/4th of the range resolution and employing a deep learning model with $2.25\times$ fewer parameters. Additionally, RadCloud utilizes a novel chirp-based approach that makes obtained point clouds resilient to rapid movements (e.g., aggressive turns or spins) that commonly occur during UAV flights. In real-world experiments, we demonstrate the accuracy and applicability of RadCloud on commercially available UAVs and UGVs, with off-the-shelf radar platforms on-board.**

I. Introduction

Light detection and ranging (lidar) sensors are often referred to as the golden standard for applications requiring highly accurate and dense 3D point clouds [1]. For example, the common Velodyne VLP-16 Puck has a horizontal angular resolution of 0.1° and a range resolution of 2 mm [2]. Such high ranging and angular resolutions make lidar sensors particularly well suited for various applications including mapping, navigation, surveying, and advanced driver assistance systems [3], [4]. However, these sensors also have poor performance in low-visibility environments like fog and smoke. Additional drawbacks include high cost, large form factors (i.e., size), and higher power consumption compared to other ranging sensors on the market. For example, the VLP-16 lidar requires a separate interface box, consumes 8 W of power during nominal operation, has a mass of 830 g, and costs $4,600 [2], [5], with drone-mounted sensors costing even more [6]. Thus, most lidars are ill-suited for resource-constrained vehicles, such as small to midsize UAVs.

On the other hand, millimeter-wave (mmWave) radio detection and ranging (radar) sensors are cheaper, smaller, lighter, and consume far less power while also providing accurate ranging information even in adverse weather and lighting conditions [7]–[9]. For example, the commercially available TI-IWR1443 mmWave radar sensor has a typical power consumption of 2 W, weighs 245 g, and the full evaluation kit costs only $398 [10], [11]. Due to the large signal bandwidth of up to 4 GHz, mmWave radars can achieve cm-level range resolutions (e.g., 4 cm for TI-IWR1443 [7], [12]).

However, radars suffer from poor angular resolution. For example, the TI-IWR1443 has a maximum azimuth resolution of 30° [13]. Thus, our goal is to enable real-time, affordable, and high-resolution sensing on resource constrained vehicles (e.g., UAVs) by using deep learning to overcome the traditional resolution limits of radar sensors.

In particular, this work introduces RadCloud, a novel *real-time* framework for efficient generation of high-resolution lidar-like 2D point clouds using low-resolution radar data for resource-constrained unmanned vehicles (e.g., UAVs and UGVs). While several recent works have explored high-resolution sensing and mapping applications using mmWave radar sensors [14]–[16], [16]–[22], they all have limitations such as only working on specific applications, relying on highly accurate position information (e.g., [16]–[20]) or not working in real-time (i.e., using offline processing). To start, [23] utilized a deep learning model to better detect real objects and filter out false radar defections to achieve higher quality point clouds. Also, [14], [15] focus on generating accurate 3D bounding boxes from radar data, but can only identify specific objects (e.g., vehicles) in the environment. By contrast, RadCloud generates a lidar-like point cloud from raw radar data for the *entire* environment, including stationary objects like walls. Similarly, RadCloud does not require position information and allows the sensing platform (i.e., the vehicle) to follow any trajectory.

Outside of traditional radar processing methods, [16], [21], [22] present methods of converting radar data into lidar-like point clouds for the purposes of indoor mapping. While [16], [21], [22] post-process radar scans of a particular scene or indoor environment to create final lidar-like point cloud mapping, RadCloud directly converts radar data frames into 2D lidar-like point clouds in *real-time*. Thus, enabling the use of the generated point clouds for other purposes like real-time navigation (in addition to mapping) and SLAM.

To the best of our knowledge, only [24] recently presented a method of directly converting raw radar data into lidar-like point clouds using a modified U-Net architecture [25].

*This work is sponsored in part by the ONR N00014-23-1-2206 and AFOSR FA9550-19-1-0169 awards, NSF CNS-1652544, CNS-2211944, and AST-2232458 grants, and the National AI Institute for Edge Computing Leveraging Next Generation Wireless Networks, Grant CNS-2112562.

The authors are with the Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708 USA (e-mail: {david.hunt, shaocheng.luo, xiao.zhang, amir.khazraei, spencer.hallyburton, tingjun.chen, miroslav.pajic}@duke.edu).

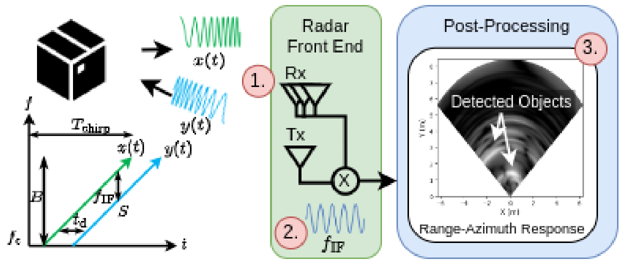

Fig. 1: Radar Signal Processing Pipeline.

However, the previously recorded radar data, sampled at the highest resolution for the radar sensor, were processed offline. On the other hand, we experimentally discovered that common UAV compute platforms (e.g., Intel NUC) cannot process the raw data from the radar at sufficient rates to support the highest resolution radar configurations from [24]. Hence, RadCloud utilizes radar data with 1/4th the maximum possible range resolution. Moreover, the use of lower-resolution radar data allows for the use of a model with less depth and 2.25× fewer parameters than the model from [24], reducing computational overhead. Despite these constraints due to the real-time radar processing, we show that RadCloud model generates sufficiently accurate (non-inferior to [24]) 2D lidar-like point clouds with 90% of predictions having errors <40 cm compared to the ground truth lidar data.

Additionally, [24] utilized the 40 most recent single-chirp frames to improve the accuracy of the generated point cloud. Yet, even with high radar sampling rates and offline processing, we find that this approach becomes significantly less accurate during rapid changes in orientation or direction (e.g., rapid vehicle turning or spinning). Thus, in contrast to such frame-based approach using the previous 2 s of sensing data, we utilize 40 radar chirps collected over a period of 8 ms. In real-world experiments, deploying RadCloud on a UAV and a UGV, we demonstrate that this chirp-based approach is much more resilient to aggressive maneuvers. To the best of our knowledge, this is the first work to implement a completely real-time framework for directly converting radar data into 2D lidar-like point clouds on resource-constrained vehicles.

This paper is organized as follows. Sec. II overviews the employed radar signal processing pipeline. Sec. III describes the system model, starting with the radar setup, before introducing the deep learning model used to generate the high-resolution point clouds. Sec. IV presents the experimental setup for RadCloud evaluation, followed by evaluation results (Sec. V), and concluding remarks (Sec. VI). Additional resources, including code and datasets are available at [26].

II. Background: Radar Signal Processing

We consider a frequency-modulated continuous wave (FMCW) radar sensor. Here, the radar's transmitter (Tx) transmits a specifically constructed signal into the environment. The signal reflects off objects which are then received by the radar's receiver (Rx). While radar sensors can detect an object's range, velocity, and angle, in this initial work we only focus on the range and angle information. The pipeline we employ is composed of three steps (Fig. 1).

Step ①: Tx and Rx chirps. For each frame, the radar transmits a series of "*chirps*" whose frequency increases linearly over time. In our framework, we transmit up to 40 chirps per frame (see Sec. III). Here, we use the following notation: S denotes the *chirp slope*, f_c denotes the *chirp start frequency*, and f_{IF} denotes the *intermediate frequency* (IF) from mixing the Tx and Rx signals. Thus, the Tx signal for a single chirp in the radar frame is given by [27]–[29]

$$x(t) = e^{j\left(2\pi f_c \cdot t + \pi S \cdot t^2\right)}. \tag{1}$$

If we assume that the environment is stationary, the time that it takes for the transmitted signal to propagate to a target at range of d and back at the speed of light (c) is given by $t_{\text{d}} = 2d/\text{c}$. Thus, the received signal can be captured as

$$y(t) = A_{\text{Rx}} \cdot e^{j\left[2\pi f_c (t - t_{\text{d}}) + \pi S (t - t_{\text{d}})^2\right]}, \tag{2}$$

where A_{Rx} denotes the received signal amplitude.

Step ②: Dechirping and IF signal generation. Next, the IF signal is obtained by mixing the Tx and Rx signals

$$s_{\text{IF}}(t) = x(t) \cdot y^*(t) = A_{\text{IF}} \cdot e^{j 2\pi f_{\text{IF}} \cdot t}, \tag{3}$$

where $f_{\text{IF}} := \frac{2S \cdot d}{c}$, $\lambda = \text{c}/f_c$ is the signal wavelength, and A_{IF} is the amplitude of the IF signal [29]. Then, the IF signal is put through a low-pass filter (typically removing all IF frequencies above 10–20 MHz) and sampled by an analog-to-digital converter (ADC) at a rate f_{samp}. For each chirp, a total of N_{Samp} I-Q samples are recorded.

Step ③: Range-Azimuth response. Using a fast Fourier transform (FFT), commonly called the "RangeFFT", the IF frequency corresponding to a target is estimated; the target's range is determined using $d_{object} = \frac{f_{\text{IF}}}{2S} \cdot c$. The range resolution ($d_{\text{res}}$ – the minimum distance between two objects detectable by a radar) and maximum range (d_{max}) are [13]

$$d_{\text{res}} = \frac{\text{c}}{2B}, \quad d_{\text{max}} = \frac{f_{\text{samp}} \cdot \text{c}}{S}. \tag{4}$$

mmWave radars use multiple receive elements to determine the angle of an object in the environment. The angular resolution for objects at boresight is defined as $\theta_{\text{res}} = 2/N_{\text{Rx}}$ (radians), where N_{Rx} is the number of Rx antennas [13]. By sampling the IF signal, a 2D FFT can be used to compute a Range-azimuth response (Fig. 1); e.g., the TI-IWR1443 radar with 4 Rx elements achieves a maximum θ_{res} of 28.6°.

III. RadCloud Design

We designed RadCloud to operate completely in real-time while also being robust to rapid movements commonly experienced by UAVs and UGVs. These design goals impacted our system design in several ways.

A. Radar Setup

Unlike all previous work, which employs offline processing of highest resolution radar data, our system processes the raw sensor data in real-time. This is an important distinction as most UAV platforms do not have a high enough computational bandwidth to support the instantaneous data rates required to process radar sensor data at the highest resolution.

We use the TI-IWR1443 radar sensor to perform sensing, and the TI-DCA1000 data capture card to send the raw data

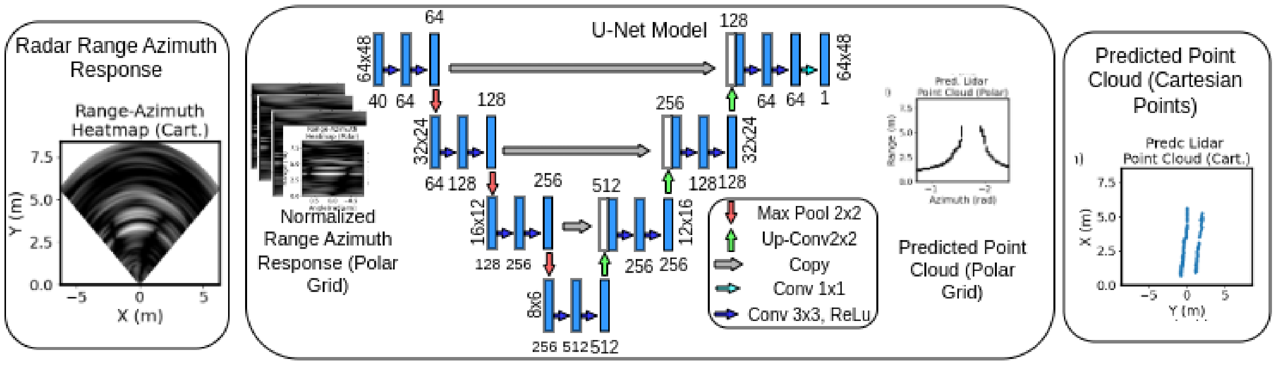
Radar Range Azimuth Response
Range-Azimuth Heatmap (Cart.)
Y (m)
X (m)
U-Net Model
64x48
40 64
64
128
32x24
64 128
128
16x12
128 256
256
512
8x6
256 512 512
12x16
256 256
256
128 128
64 64 1
Normalized Range Azimuth Response (Polar Grid)
Max Pool 2x2
Up-Conv2x2
Copy
Conv 1x1
Conv 3x3, ReLu
Pred. Lidar Point Cloud (Polar)
Range (m)
Azimuth (rad)
Predicted Point Cloud (Polar Grid)
Predicted Point Cloud (Cartesian Points)
Predc Lidar Point Cloud (Cart.)
X (m)
Y (m)

Fig. 2: RadCloud framework overview.

to the host [11], [30], [31]. This platform has been used in most previous works involving mmWave radar (e.g., [16], [21], [22], [24], [32]). Here, the radar sends complex-valued samples of the IF signal, $s_{\text{IF}}(t)$, captured at rate f_{samp}, where each sample is 4-Byte (16-bit integer for real and 16-bit integer for complex). Thus, operating the radar at the maximum f_{samp} of 18.75 MSa/s requires the platforms to support an instantaneous data rate of 2.4 Gbps [11], [30], [31].

Unfortunately, many UAV platforms do not have sufficiently high computation bandwidth to support data sent at such high instantaneous rates due to limited computational resources. Also, radar configurations with $N_{\text{Samp}} \geq 90$ require multiple packets per chirp (due to the maximum Ethernet packet size of 1462 B [31]). For our platform, we empirically decided to operate the radar with $f_{\text{samp}} = 2$ MSa/s (instantaneous data rate of 256 Mbps) and $N_{\text{Samp}} = 64$[1] to ensure real-time data processing and avoid packet losses.

Radar Configuration. Given the constraints imposed by the considered UAV platforms, we selected a radar configuration that maximized the range resolution (d_{res}) while achieving a maximum range (d_{max}) of roughly 10 m. Thus, our final configuration utilized chirps with $S = 35$ MHz/µs, $f_{\text{samp}} = 2$ MSa/s, and $N_{\text{Samp}} = 64$, achieving a chirp bandwidth (B) of 1.12 GHz. From (4), we achieve a $d_{\text{res}} =$ 13.3 cm and $d_{\text{max}} = 8.56$ m. Compared with previous works (e.g., [24] that used the maximum $B = 4$ GHz with $N_{\text{Samp}} =$ 256 to achieve $d_{\text{res}} = 3.7$ cm and $d_{\text{max}} = 9.59$ m), the real-time data streaming constraints restrict our system to utilizing radar data with roughly 1/4th the range resolution. Finally, we note the radar configuration only performs 2D sensing in light of the sensor's 6 dB elevation beamwidth of $\pm 20°$.

B. Deep Learning (DL) Model

We develop a DL model based on the U-Net architecture [25], which takes in a normalized Range-Azimuth response from the radar (in polar coordinates) and outputs a quantized grid representing the lidar-like point cloud (in polar coordinates). The output point cloud is then converted into the 2D cartesian coordinates that can be used for navigation, mapping, or other point cloud detection algorithms. Fig. 2 presents an overview of the RadCloud framework.

[1]Empirical observations showed a significant number of dropped Ethernet packets for configurations requiring multiple Ethernet packets per chirp.

Model Input. For each radar chirp, we compute a complex-valued range-azimuth response (Step 3 from Sec. II) with 64 range bins and 64 azimuth bins[2]. To convert the response into a format that can be used by a DL model, we start by taking the magnitude of the complex data, applying a threshold to filter out very weak reflections[3], and normalize the response to between 0 and 1. While the field of view of the radar is theoretically $\pm 90°$ due to the Rx element spacing, we only use parts of the Range-Azimuth response that are within $\pm 50°$ as this is the horizontal 6 dB beamwidth of the radar, and the radar's angular resolution significantly decreases at angles outside of this field-of-view [13], [30].

Compared to [24], which utilized 2 s worth of previous frames to improve model accuracy, we empirically decided to employ 40 chirps from a single radar frame, requiring only 8 ms of total radar sensing time, i.e., a 250× reduction. Thus, the final input to the model is a $40 \times 64 \times 48$[4] tensor corresponding to the 40 normalized range azimuth responses.

Our chirp-based approach is particularly important because a radar's view of the environment changes rapidly when the platform experiences rapid movement (e.g., a vehicle spinning/turning quickly or moving at high speeds). For the previous frame-based approach [24], this causes the scene captured in the first few frames to vary drastically compared to the scene captured in the last few frames. As we show in Sec. V, the performance of these frame-based models noticeably degrades when moving rapidly because the sensed environment can change significantly over several frames in such cases. Thus, we show that our chirp-based approach *significantly improves robustness to aggressive maneuvers*.

Model Output. We take three steps when pre-processing the ground-truth lidar data used to train and evaluate our model's performance. To start, we filter the lidar point cloud so that it has the same azimuth field of view as the input radar data. Next, we obtain a 2D lidar point cloud by only keeping points with $-20\,\text{cm} \leq z \leq 10\,\text{cm}$ to filter

[2]The 64 azimuth bins are achieved by zero-padding the azimuth FFT's input to 64 bins to increase the smoothness of the generated response. The resulting azimuth FFT has an FFT bin resolution (different from θ_{res}) of $1.8°$ at boresight and $15°$ at $90°$ off of boresight [13], [30].

[3]We filter out all reflections that are 45 dB less than the maximum reflected signal power as such reflections are often just noise.

[4]The reduction of the azimuth dimmension from 64 to 48 occurs due to the narrowing the radar's FOV from $\pm$ 90° to $\pm 50°$.

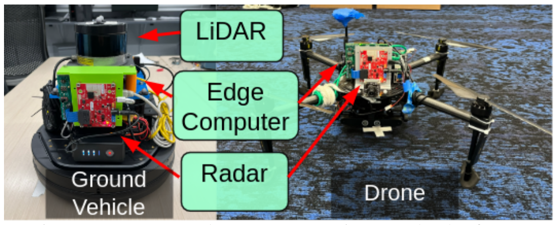

Fig. 3: UGV and UAV experimental platforms.

out undesired ground detections and in light of the radar sensor's 2D configuration. Here, we match the input and output dimensions by converting the Cartesian point cloud to polar coordinates and then quantizing the points into a 64×48 polar grid with a horizontal resolution of 2.08° and a range resolution of 13.3 cm. Thus, to obtain a lidar-like point cloud from the model's prediction, we convert the prediction grid into a set of Cartesian points.

Model Architecture. We use a simplified U-net architecture [25] to generate the higher resolution point clouds. The primary benefit of this architecture is its encoder-decoder structure. In our case, the encoder progressively downsamples the input data while capturing key context and feature information from the input radar data. Then, the decoder outputs the higher resolution point cloud by generating features from the encoded data while also preserving spatial information through the use of skip connections [25].

The final model architecture is shown in Fig. 2. To ensure our model's real-time performance on resource constrained platforms, we implemented a simplified model architecture with fewer layers and less depth compared to the model from [24]; our model utilizes $\sim$7.7 M parameters compared to the $\sim$17.5 M parameters used by [24], demonstrating a $2.25\times$ reduction in the number of parameters. As we show in the following sections, this simpler model allows us to achieve real-time frame rates, even on CPU-only machines.

Loss Function. As shown in [24], utilizing a combination of Binary Cross Entropy (BCE) loss and Dice loss is particularly effective when converting radar data to lidar-like point clouds. Here, the BCE loss seeks to force each predicted pixel to be as close to the actual value as possible while the added Dice loss helps to make the predicted features sharper. Similar to [24], we weighted the BCE loss to be 0.9 while weighting the Dice loss by a factor of 0.1.

IV. Experiments

A. Experimental Platform

We demonstrate the real-world capability of RadCloud by implementing it on a common vehicle platform.

Radar. We utilize the commercially available TI-IWR1443 radar to sense the environment and the TI-DCA1000 data capture card to stream the raw radar data to the edge compute platform in real-time [11], [30], [31]. Here, we implemented a ROS-compatible real-time streaming interface in Python to obtain data from the TI-IWR1443 via the TI-DCA1000.

Lidar. To obtain ground truth information, we use the popular VLP-16 Puck lidar sensor [2], [5]. Here, we ensure a consistent extrinsic calibration between the radar and lidar sensors by mounting the radar sensor 7 cm below and 10 cm in front of the lidar sensor, enabling the model to 'learn' the relative position of the radar with respect to the lidar.

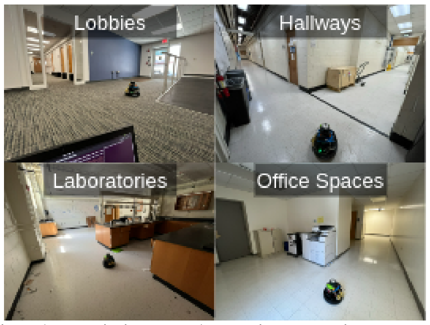

Fig. 4: Training and Testing Environments.

Vehicle Computer. We implement our entire real-time framework on the NUC7i5BNH as our vehicle computing platform [33]. Unlike previous works (e.g., [24]), which use powerful Jetson platforms with an integrated GPU, our platform only has a dual-core 3.4 GHz Intel i5 CPU and no GPU. *We highlight that we are the first to implement a completely real-time system, including streaming data from radar to the NUC7i5BNH, range-azimuth response computation, and generating high-resolution 2D lidar-like point clouds.* While we operate our system at 10 frames per second, we achieved average frame rates above 15 frames per second when testing our full pipeline on the NUC7i5BNH. Overall, the full ROS-compatible framework is implemented in over 7,000 lines of code, which is responsible for the real-time capturing of radar data and its conversion into lidar-like 2D point clouds.

Robotic Platforms. We mount the RadCloud platform on a Kobuki ground vehicle and a DJI Matrice 100 drone [34], [35] (Fig. 3). While we were able to mount the VLP-16 lidar onto the ground vehicle, we were unable to mount it on the drone due to the size and weight limitations on the drone. Thus, we utilize the ground vehicle platform to assess model performance and the drone-based platform to demonstrate the feasibility of our framework in airborne environments.

B. Experimental Setup

Training, Validation, and Test Datasets. For training, validation, and initial testing of the RadCloud model, we capture time-synchronized radar and lidar frames, sampled at a frame rate of 10 Hz, across 7 different environments including laboratories, corridors, and building lobbies (Fig. 4). To improve model robustness, we also drive the ground vehicle along various trajectories including turns, spinning, and straight-line movement at a range of angular and linear velocities. We recorded a total of 87,476 samples for training, validation, and testing; we used 66,609 samples for training, 11,755 samples for validation and parameter tuning, and 9,112 samples for testing. The test dataset was recorded independently from the training and validation datasets, and it included unique trajectories. Thus, we were able to assess our model's performance when operating in the "same" environment.

Unseen Environment Test Dataset. In addition to the 7 environments used for training, validation, and initial testing, we also recorded an additional 4,767 samples in 3 unseen environments that the model had not previously been trained on. We used the results of this "unseen environment" dataset to assess the model's performance when operating in unfamiliar environments, similar to how a UAV or a UGV may be used to map or navigate in unknown environments.

Rapid Movement Test Dataset. Finally, we record a third dataset specifically to evaluate our model's resiliency to rapid movements (e.g., spinning, fast turns, and high speed movements) commonly encountered by UAVs and UAGs. While both the training and test set do include some aggressive movements, the majority of radar and lidar frames were recorded at relatively slower speeds. Thus, we recorded an additional 784 samples while driving the UGV along different trajectories at maximum speed and rotational velocity, to enable evaluating the model's performance in such situations.

C. Evaluation Metrics

We utilize the commonly used Chamfer and Modified Hausdorff metrics to evaluate the accuracy of RadCloud model's predicted point cloud compared to the ground truth point cloud obtained from the lidar [36]–[38]. Here, we define the Chamfer distance (CD) as

$$\mathrm{CD}(S_1, S_2) = \frac{1}{2|S_1|}\sum_{x\in S_1}\min_{y\in S_2} d(x,y) + \frac{1}{2|S_2|}\sum_{y\in S_2}\min_{x\in S_1} d(x,y), \tag{5}$$

and Modified Hausdorff distance (MHD) as

$$\mathrm{MHD}(S_1, S_2) = \max\left\{\underset{x\in S_1}{\mathrm{med}}\min_{y\in S_2} d(x,y), \underset{y\in S_2}{\mathrm{med}}\min_{x\in S_1} d(x,y)\right\},$$

where $d(x,y)$ denotes the Euclidean distance i.e., $||x-y||_2^2$.

D. Comparison to Baseline

We were unable to compare our model's performance with the model from [24] as the input and output data dimensions for RadCloud model are different than the one used by [24]. Thus, we train two additional models that utilize the previous frame-based approach to compare our model with such a 'baseline'. Specifically, we train a model that uses the previous 20 (single-chirp) frames and another model that uses the previous 40 (single-chirp) frames.

V. Results

In this section, we present the results from our experimental evaluations, comparing our with the baseline models.

A. Performance in Same and Unseen Environments

Fig. 5 presents the CDFs of the CD and MHD for the RadCloud model and the two baseline models, operating in the previously seen (e.g., same as training) environments and unseen environments. Table I and Table II summarize the key metrics for each distribution. The results show that our chirp-based approach is nearly as good as the previous frame-based approaches at generating high-resolution point clouds from low-resolution radar data. This is further supported by Fig. 6 showing a predicted point cloud from the RadCloud model.

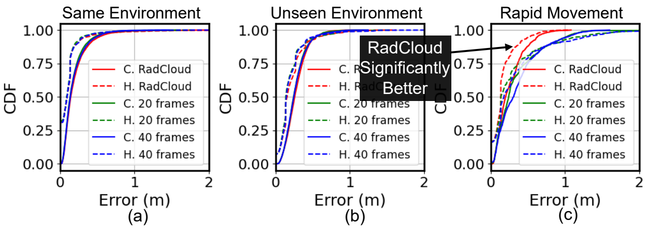

Fig. 5: Error distributions for (a) same environment, (b) unseen environments, and (c) during rapid movements

TABLE I: Error Comparison - Same Environment

Metric	Units	RadCloud	20 frames	40 frames
Cham. (Mean)	m	0.20	**0.18**	**0.18**
Cham. (Median)	m	0.14	**0.13**	0.14
Cham. (90%)	m	0.40	**0.36**	0.38
MHaus. (Mean)	m	0.12	**0.11**	**0.11**
MHaus. (Median)	m	**0.09**	**0.09**	**0.09**
MHaus. (90%)	m	0.23	**0.20**	0.23

TABLE II: Error Comparison - Unseen Environment

Metric	Units	RadCloud	20 frames	40 frames
Cham. (Mean)	m	0.27	0.27	**0.26**
Cham. (Median)	m	0.26	**0.24**	**0.24**
Cham. (90%)	m	0.46	**0.45**	**0.45**
MHaus. (Mean)	m	0.22	**0.19**	0.20
MHaus. (Median)	m	0.15	**0.09**	0.14
MHaus. (90%)	m	0.41	**0.40**	**0.40**

TABLE III: Error Comparison - Aggressive Maneuvers

Metric	Units	RadCloud	20 frames	40 frames
Cham. (Mean)	m	**0.26**	0.35	0.37
Cham. (Median)	m	**0.20**	0.24	0.31
Cham. (90%)	m	**0.53**	0.83	0.81
MHaus. (Mean)	m	**0.17**	0.32	0.34
MHaus. (Median)	m	**0.13**	**0.13**	0.17
MHaus. (90%)	m	**0.39**	0.82	0.87

As shown, our model's output is almost identical to the ground truth lidar point cloud, demonstrating that the RadCloud model is well suited for converting low-resolution radar range-azimuth responses to high-resolution lidar-like 2D point clouds. We also highlight how our model does a good job of capturing complex shapes in the environment like various corner shapes. The results also show that the RadCloud model still generates accurate point clouds even in unseen environments, enabling its use on UAVs and UAGs to map or navigate unseen environments. Finally, we highlight the accuracy of our model's predictions with over 90% of generated point clouds having a CD less than 46 cm and a MHD less than 41 cm when compared to the ground truth lidar point cloud, even in unseen environments.

B. Rapid Movement Performance

Fig. 5(c) presents the CDFs of the CD and MHD for our model and the two baseline models during aggressive maneuvers, whereas Table III summarizes the key metrics for each distribution. Compared to the other scenarios, the performance of the frame-based models noticeably degrades

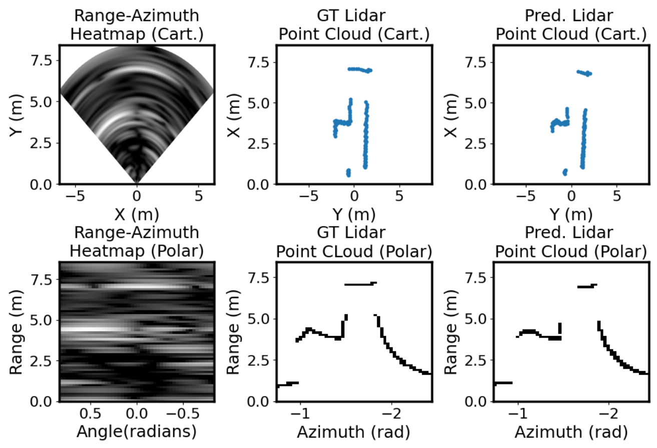

Fig. 6: Input radar data, ground truth point cloud, and predicted point cloud for nominal operation. The bottom row shows the format at the inputs/outputs of the model. The top row shows the data in a Cartesian format.

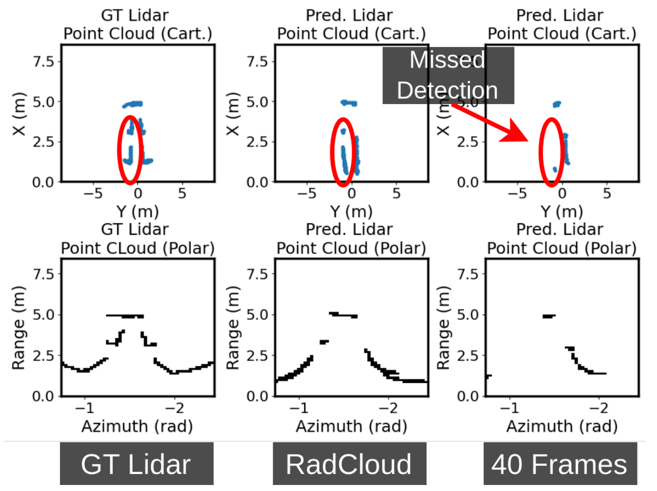

Fig. 7: Generated point clouds during rapid movements.

in cases of fast movements. For example, the 20 frames model went from 90% of predictions having a CD less than 36 cm and an MHD less than 20 cm, to 90% of predictions having a CD less than 83 cm and an MHD less than 82 cm. By contrast, RadCloud only experiences slight increases in both CD and MHD while also performing noticeably better than the previous frame-based models.

As presented in Fig. 7, RadCloud's model still manages to detect the main features of the environment while the 40 frames model fails to detect large features in the environment due to the rapid movements. Overall, these results show that our model is significantly more resilient to aggressive maneuvers compared to the previous frame-based approaches.

C. UAV Case Study

As discussed in Sec. IV, we mounted the radar and a NUC platform onto a commercially available DJI Matrice 100 drone [35]. While we were not able to obtain ground truth information due to the size and weight of the lidar sensor, this case study demonstrates the feasibility and practicality of the RadCloud real-time framework. We flew the drone inside the

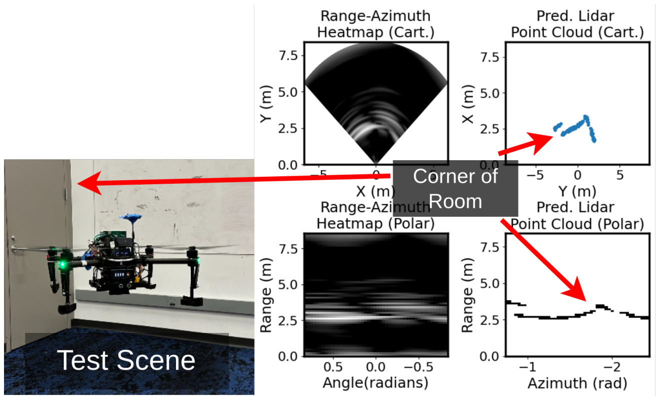

Fig. 8: Performance on a Drone Based Platform.

environment pictured in Fig. 8. Here, we highlight that the model was not trained on this environment nor did we train the model on a drone platform. This is particularly notable as the radar experiences different dynamics (e.g., vibrations) on the drone, and the drone's propellers also add additional noise and interference to the radar data.

Thus, this case study also demonstrates the RadCloud model's performance in unseen environments and on different platforms. Fig. 8 presents one of the predicted point clouds generated using our model (a more complete video is available at [26]).As shown, our system still does a good job of identifying the major features (e.g., walls and corners of the room), even in unseen environments and on different platforms. Combined, these results demonstrate the feasibility and practicality of the RadCloud platform. Thus, we demonstrate ***a real-time framework for directly converting low resolution radar data to 2D lidar-like point clouds, which can be used for mapping, navigation, and other purposes on UAVs***.

VI. Conclusion

In this work, we have presented RadCloud a *real-time* framework for directly deriving high-resolution lidar-like 2D point clouds from low-resolution radar frames on resource-constrained platforms commonly used in unmanned aerial and ground vehicles; the high-resolution of the point clouds enables their use in accurate environmental mapping, navigation in unknown environments, as well as other robotics tasks. Since existing methods for high-resolution sensing from radar data cannot be used on resource-constrained platforms, RadCloud has overcome the challenges presented by these platforms by utilizing a radar configuration with 1/4th the range resolution and deep learning model with 2.25$\times$ fewer parameters. Further, we have utilized a novel chirp-based approach making generated point clouds more resilient to aggressive turns, spins, and other rapid movements commonly experienced during UAV and UGV operations. Finally, we have demonstrated the accuracy and applicability of RadCloud on commonly used UAVs and UGVs with commercially available radar platforms on board, where we have achieved average frame rates of 15fps even when operating on CPU-only platforms with limited computational power.

References

[1] P. Dong and Q. Chen, *LiDAR remote sensing and applications*. CRC Press, 2017.

[2] V. LiDAR, "VLP-16 User Manual," Oct. 2018. [Online]. Available: https://velodynelidar.com/wp-content/uploads/2019/12/63-9243-Rev-E-VLP-16-User-Manual.pdf

[3] S. Royo and M. Ballesta-Garcia, "An Overview of Lidar Imaging Systems for Autonomous Vehicles," *Applied Sciences*, vol. 9, no. 19, p. 4093, Sept. 2019. [Online]. Available: https://www.mdpi.com/2076-3417/9/19/4093

[4] J. S. Payne, "Autonomous interior mapping robot utilizing lidar localization and mapping," Ph.D. dissertation, Monterey, CA; Naval Postgraduate School, 2020.

[5] "Puck." [Online]. Available: https://store.clearpathrobotics.com/products/puck

[6] "Zenmuse L1." [Online]. Available: https://enterprise.dji.com/zenmuse-l1

[7] Keysight, "How Millimeter Wave Automotive Radar Enhances Advanced Driver Assistance Systems (ADAS) and Autonomous Driving," Keysight Technologies, Tech. Rep., 2020. [Online]. Available: https://www.keysight.com/us/en/assets/7018-06176/white-papers/5992-3004.pdf

[8] A. Benjamin, "Imaging radar: one sensor to rule them all," Texas Instruments, Tech. Rep., 2019. [Online]. Available: https://e2e.ti.com/blogs_/b/behind_the_wheel/posts/imaging-radar-using-ti-mmwave-sensors

[9] M. Gardill, "Automotive Radar - An Overview on State-of-the-Art Technology," 2019. [Online]. Available: https://www.youtube.com/watch?v=P-C6_4ceY64&ab_channel=IEEEMicrowaveTheoryandTechnologySociety

[10] Mouser, "IWR1443Boost." [Online]. Available: https://www.mouser.com/ProductDetail/Texas-Instruments/IWR1443BOOST?qs=5aG0NVq1C4wT7gyvvDbMRw%3D%3D

[11] T. Instruments, "IWR1443 Single-Chip 76- to 81GHz mmWave Sensor," Oct. 2018. [Online]. Available: https://www.ti.com/lit/gpn/iwr1443

[12] K. Ramasubramanian, K. Ramaiah, and A. Aginskiy, "Moving from Legacy 24 GHz to State-of-the-Art 77-GHz Radar," Texas Instruments, Tech. Rep., 2017. [Online]. Available: https://www.ti.com/lit/wp/spry312/spry312.pdf

[13] S. Rao, "Introduction to mmwave Sensing: FMCW Radars." [Online]. Available: https://training.ti.com/sites/default/files/docs/mmwaveSensing-FMCW-offlineviewing_0.pdf

[14] S. Madani, J. Guan, W. Ahmed, S. Gupta, and H. Hassanieh, "Radatron: Accurate Detection Using Multi-resolution Cascaded MIMO Radar," in *Comput. Vision – ECCV'22*, S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds. Cham: Springer Nature Switzerland, 2022, vol. 13699, pp. 160–178, series Title: Lecture Notes in Computer Science. [Online]. Available: https://link.springer.com/10.1007/978-3-031-19842-7_10

[15] M. Meyer, G. Kuschk, and S. Tomforde, "Graph Convolutional Networks for 3D Object Detection on Radar Data," in *2021 IEEE/CVF Int. Conf. on Comput. Vision Workshops (ICCVW'21')*. Montreal, BC, Canada: IEEE, Oct. 2021, pp. 3053–3062. [Online]. Available: https://ieeexplore.ieee.org/document/9607427/

[16] K. Qian, Z. He, and X. Zhang, "3D Point Cloud Generation with Millimeter-Wave Radar," *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 4, no. 4, pp. 1–23, Dec. 2020. [Online]. Available: https://dl.acm.org/doi/10.1145/3432221

[17] M. E. Yanik, D. Wang, and M. Torlak, "Development and Demonstration of MIMO-SAR mmWave Imaging Testbeds," *IEEE Access*, vol. 8, pp. 126 019–126 038, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9136646/

[18] X. Gao, S. Roy, and G. Xing, "MIMO-SAR: A Hierarchical High-Resolution Imaging Algorithm for mmWave FMCW Radar in Autonomous Driving," *IEEE Trans. on Veh. Technol.*, vol. 70, no. 8, pp. 7322–7334, Aug. 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9465646/

[19] A. Sengupta, F. Jin, R. A. Cuevas, and S. Cao, "A Review of Recent Advancements Including Machine Learning on Synthetic Aperture Radar using Millimeter-Wave Radar," in *2020 IEEE Radar Conf. (RadarConf'20)*. Florence, Italy: IEEE, Sept. 2020, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/9266501/

[20] E. Schreiber, A. Heinzel, M. Peichl, M. Engel, and W. Wiesbeck, "Advanced Buried Object Detection by Multichannel, UAV/Drone Carried Synthetic Aperture Radar," 2019.

[21] C. X. Lu, S. Rosa, P. Zhao, B. Wang, C. Chen, J. A. Stankovic, N. Trigoni, and A. Markham, "See Through Smoke: Robust Indoor Mapping with Low-cost mmWave Radar," May 2020, arXiv:1911.00398 [eess]. [Online]. Available: http://arxiv.org/abs/1911.00398

[22] P. Cai and S. Sur, "MilliPCD: Beyond Traditional Vision Indoor Point Cloud Generation via Handheld Millimeter-Wave Devices," *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 6, no. 4, pp. 1–24, Dec. 2022. [Online]. Available: https://dl.acm.org/doi/10.1145/3569497

[23] Y. Cheng, J. Su, M. Jiang, and Y. Liu, "A novel radar point cloud generation method for robot environment perception," *IEEE Trans. on Robot.*, vol. 38, no. 6, pp. 3754–3773, 2022.

[24] A. Prabhakara, T. Jin, A. Das, G. Bhatt, L. Kumari, E. Soltanaghai, J. Bilmes, S. Kumar, and A. Rowe, "High Resolution Point Clouds from mmWave Radar," in *2023 IEEE Int. Conf. on Robot. and Automat. (ICRA'23)*. London, United Kingdom: IEEE, May 2023, pp. 4135–4142. [Online]. Available: https://ieeexplore.ieee.org/document/10161429/

[25] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," May 2015, arXiv:1505.04597 [cs]. [Online]. Available: http://arxiv.org/abs/1505.04597

[26] "RadCloud." [Online]. Available: https://sites.google.com/view/radcloudduke

[27] S. Alland, W. Stark, M. Ali, and M. Hegde, "Interference in Automotive Radar Systems: Characteristics, Mitigation Techniques, and Current and Future Research," *IEEE Signal Process. Mag.*, vol. 36, no. 5, pp. 45–59, Sept. 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8828037/

[28] C. Jiang, J. Guo, Y. He, M. Jin, S. Li, and Y. Liu, "mmVib: micrometer-level vibration measurement with mmwave radar," in *Proc. of the 26th Annu. Int. Conf. on Mobile Comput. and Netw. (MobiCom'20)*. London United Kingdom: ACM, Sept. 2020, pp. 1–13. [Online]. Available: https://dl.acm.org/doi/10.1145/3372224.3419202

[29] Y. Wang, W. Wang, M. Zhou, A. Ren, and Z. Tian, "Remote Monitoring of Human Vital Signs Based on 77-GHz mm-Wave FMCW Radar," *Sensors*, vol. 20, no. 10, p. 2999, May 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/10/2999

[30] T. Instruments, "IWR1443 Evaluation Module (IWR1443BOOST) mmWave Sensing Solution User's Guide (Rev. D).pdf," May 2020.

[31] ——, "DCA1000EVM Data Capture Card User's Guide (Rev. A).pdf," May 2019.

[32] D. Hunt, K. Angell, Z. Qi, T. Chen, and M. Pajic, "MadRadar A Black-Box Physical Layer Attack Framework on mmWave Automotive FMCW Radars," in *The 2024 Network and Distributed System Security Symp. (NDSS)*, Can Diego, CA, Feb. 2024.

[33] Intel, "Intel NUC Kit NUC7i5BNH," Sept. 2023. [Online]. Available: https://ark.intel.com/content/www/us/en/ark/products/95067/intel-nuc-kit-nuc7i5bnh.html

[34] Kobuki, "Kobuki-UserGuide.pdf," Jan. 2016. [Online]. Available: http://file.ncnynl.com/ros/Kobuki-UserGuide.pdf

[35] DJI, "Matrice 100 Specs," Sept. 2023. [Online]. Available: https://www.dji.com/matrice100

[36] A. Bell, B. Chambers, and H. Butler, "Chamfer," Sept. 2023. [Online]. Available: https://pdal.io/en/latest/apps/chamfer.html

[37] D. Watkins, "Chamfer Distance API," Sept. 2023. [Online]. Available: https://github.com/DavidWatkins/chamfer_distance

[38] M.-P. Dubuisson and A. Jain, "A modified Hausdorff distance for object matching," in *Proc. of 12th Int. Conf. on Pattern Recognition*, vol. 1. Jerusalem, Israel: IEEE Comput. Soc. Press, 1994, pp. 566–568. [Online]. Available: http://ieeexplore.ieee.org/document/576361/