Towards Foundational Models for Single-Chip Radar

Tianshu Huang[12] Akarsh Prabhakara[3] Chuhan Chen[1] Jay Karhade[1]
Deva Ramanan[1] Matthew O'Toole[1] Anthony Rowe[12]
[1]Carnegie Mellon University [2]Bosch Research [3]University of Wisconsin–Madison
{tianshu2, chuhanc, jkarhade, deva, motoole2, agr}@andrew.cmu.edu, akarsh@cs.wisc.edu

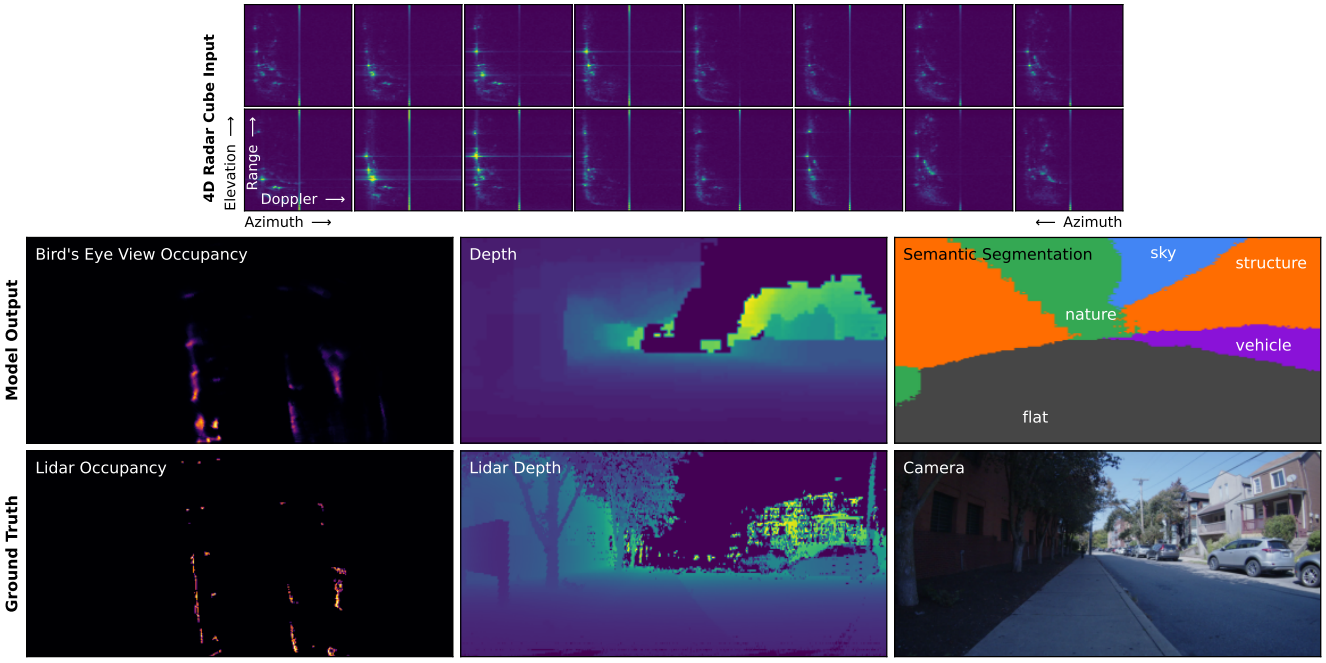


Figure 1. **Using only 4D range-Doppler-azimuth-elevation data from a radar with a 3x4 antenna array – equivalent to a 0.26-megapixel camera,** we trained (separate) radar transformers on 24 hours (1M radar-lidar-camera samples) of data to predict Bird's Eye View 2D occupancy (left), 3D occupancy (center), semantic segmetation (right), and Ego-Motion.

Abstract

mmWave radars are compact, inexpensive, and durable sensors that are robust to occlusions and work regardless of environmental conditions, such as weather and darkness. However, this comes at the cost of poor angular resolution, especially for inexpensive single-chip radars, which are typically used in automotive and indoor sensing applications. Although many have proposed learning-based methods to mitigate this weakness, no standardized foundational models or large datasets for the mmWave radar have emerged, and practitioners have largely trained task-specific models from scratch using relatively small datasets.

In this paper, we collect (to our knowledge) the largest available raw radar dataset with 1M samples (29 hours) and train a foundational model for 4D single-chip radar, which can predict 3D occupancy and semantic segmentation with quality that is typically only possible with much higher resolution sensors. We demonstrate that our Generalizable Radar Transformer (GRT) generalizes across diverse settings, can be fine-tuned for different tasks, and shows logarithmic data scaling of 20% per $10\times$ data. We also run extensive ablations on common design decisions, and find that using raw radar data significantly outperforms widely-used lossy representations, equivalent to a $10\times$ increase in training data. Finally, we roughly estimate that $\approx$100M samples (3000 hours) of data are required to fully exploit the potential of GRT.

1. Introduction

As a compact, inexpensive [27], and robust solid-state sensor, mmWave radars are ideal for sensing applications ranging from simple automatic door openers [58] to autonomous

drones [11] or vehicles [53, 62]. mmWave radars are rich sensors which can directly measure range and velocity while capturing a unique range of material properties [18]; however, this comes at the cost of poor angular resolution typically on the order of 15° – orders of magnitude worse than cameras or lidars [59].

Radar data are typically processed into radar point clouds (Fig. 2) derived using Constant False Alarm Rate (CFAR) peak detectors [36, 52] combined with Angle-of-Arrival estimation techniques [60]. However, this is a substantially lossy process: while raw radar data suffers from unique noise patterns such as "bleed" and side lobes [25], weak reflectors and other signals can be hidden in this noise, which would ordinarily be filtered out.

On the other hand, raw spectrum (4D range-Doppler-azimuth-elevation data cubes [50]) can be unintuitive and difficult to interpret compared to lidar point clouds or camera images, and include properties such as specularity and Doppler which lack straight-forward Cartesian interpretations [18]. As such, many machine learning methods have been proposed [24, 45, 69] to exploit 4D radar data from single-chip radars, achieving remarkable performance on 2D scene understanding tasks. However, due to the dominance of CFAR point clouds in radar processing, as well as the high data rate of raw mmWave radar data, most radar toolchains only process point-cloud data. Tooling for raw I/Q (in-phase/quadrature) data is often brittle, poorly documented, and largely unsupported by radar vendors, severely limiting the availability and scale of both raw mmWave datasets and the models which operate on raw data.

To rectify this limitation, we develop an open-source toolchain and associated large dataset specifically for 4D mmWave radar data. Training a radar-to-lidar model and fine-tuning for a range of other tasks, we demonstrate the surprising effectiveness of mmWave radar models trained at scale (Fig. 1). Going further, just as large foundational models [4] have greatly accelerated the pace of innovation in computer vision and natural language processing, we believe that a foundational model for raw mmWave radar trained at even larger scale could similarly supercharge the advancement of radar sensing techniques.

Contributions In this paper, we develop a full stack[1] for collecting data, training, and evaluating a transformer for 4D single-chip radar to quantify both the potential costs and benefits of training a foundational model at scale. To summarize our contributions:

(1) We develop a compact, lightweight multimodal data collection system (Sec. 3.1) capable of collecting synchronized raw radar, Lidar, and camera data which can be operated as a handheld device. Our system can be easily replicated using off-the-shelf components, 3D printed parts, and our open-source software.

(2) Using this data collection system, we collect a dataset, I/Q-1M (*One Million IQ Frames*), consisting of 29 hours of data – 8× longer than the next largest publicly available raw radar dataset – split between indoor, outdoor-handheld, and bike-mounted settings, each with different radar configurations (Sec. 3.2).

(3) Finally, using our dataset, we train a Generalizable Radar Transformer (GRT), which can output depth maps and segmentation images with quality which is typically only possible with much higher resolution radars. Using GRT, we then run ablations on common design choices (Sec. 5.1), quantify the scalability of GRTs with increasing dataset and model size (Sec. 5.2, 5.4), and demonstrate that our GRT can be readily fine-tuned for other tasks and settings (Sec. 5.3), including obtaining state-of-the-art performance on the Coloradar [23] dataset with 30-minutes of fine-tuning.

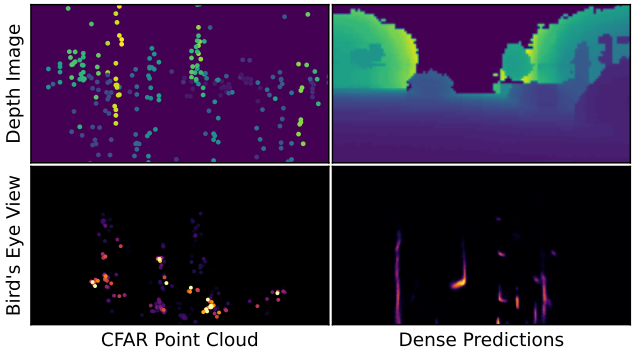


Figure 2. While a transformer can generate Lidar-like depth and bird's eye view images, traditional CFAR point clouds are noisy, and have poor angular resolution – especially in the elevation axis.

Key Findings We summarize our key findings as follows:

- **Radar models can generalize** to different settings and radar configurations (Sec. 5.2), as well as across objectives (with some fine-tuning). This suggests great potential for a cross-domain foundational model to improve and accelerate the development of new radar models.
- **Using raw data yields outsized performance gains**, equivalent to more than a 10× increase in training data (Sec. 5.1). While existing datasets largely focus on CFAR point clouds or other processed representations, we believe that more emphasis should be placed on making raw data available for research.
- **Existing mmWave radar datasets are vastly undersized**. 24 hours of training data is not enough to saturate even a 4M parameter model! Our analysis suggests that **at least 100× more data** is required to exploit the full potential of radar transformer models (Sec. 5.4).

[1] Our data collection system, dataset, code, and model can be found via our project site: https://wiselabcmu.github.io/grt/.

Figure 3. **Our data collection rig** in its handheld (right) and bike-mounted (left) configurations; see App. A for additional images.

Limitations Despite its size compared to previous datasets, I/Q-1M is quite small compared to the datasets used to train modern vision transformers, which can exceed billions of samples [68]. I/Q-1M also only includes daylight conditions and fair weather, and lacks the scale to capture "edge cases" that would be represented in a larger dataset [22]. Finally, since I/Q-1M uses a single type of radar, we cannot evaluate generalization across different antenna configurations — only radar configurations.

2. Related Work

4D Solid State Radar Excluding mechanical radars, which perceive the world as lidar-like 2D heatmaps [3, 5, 54] via a rotating antenna, solid-state mmWave radars operate by transmitting and receiving a sequence of frequency-modulated "chirps" from an array of transmit (TX) and receive (RX) antenna [19]; this data is typically (losslessly) transformed to a 4D range-Doppler-azimuth-elevation data cube using a 4D FFT [50], whose resolutions are constrained by bandwidth, form factor, and the integration window. We focus on single-chip radars which have compact form factors – and thus poor angular resolution.

Learning and Datasets for 4D Radar Most radar processing methods use point clouds extracted from the 4D cubes [36, 52] as inputs [1, 7, 32, 44, 55, 56], allowing them to re-use popular Lidar architectures or even pre-trained models such as PointNet [46]. However, since this discards much of the information contained in a 4D radar cube, many competing approaches propose to directly interpret the 4D radar cube using methods and architectures such as feedforward convolutional architectures [9, 38, 43, 49, 69], multi-view convolutional networks across different tensor axes [13, 34, 40], U-Nets [26, 39, 45], diffusion models [70], and transformers [2, 15, 21, 21].

These prior machine learning-based approaches rely on publicly available datasets with 4D radar data, including from both cascaded [42, 49] and single-chip [23, 29, 69] radars; however, existing datasets are relatively small, with 3.6-hour RaDICal [29] and 2.4-hour Coloradar [23] as the largest (Table 1). Due to the success of powerful but data-hungry [68] transformer models [61] in computer vision [12], we believe that limited data availability imposes a substantial bottleneck on learning for 4D radar.

Table 1. **Comparison with other single-chip mmWave radar datasets;** a *frame* refers to the number of unique radar-sensor samples. For comparisons with datasets using other types of radar, see App. A.3. Our dataset is significantly larger than previous single-chip radar datasets, enabling us to explore scaling up models.

Dataset	4D Data Cube	Dataset Size
I/Q-1M (Ours)	**Yes**	**29 hours (1M frames)**
MilliPoint [7]	No (3D Points)	6.3 Hours (545k frames)
RaDICal [29]	**Yes**	3.6 Hours (394k frames)
CRUW [63]	No (2D Map)	3 hours (400k frames)
Coloradar [23]	**Yes**	2.4 hours (82k frames)
RadarHD [45]	**Yes**	200k frames
CARRADA [41]	No (3D Cube)	21 Minutes (13k frames)
RADDet [69]	**Yes**	10k frames

High-resolution Imaging from Low-Resolution Radar Due to the low angular resolution of single-chip radars, extracting high-resolution angular information can be challenging; as a result, prior work focuses on recovering 2D spatial information [15, 45, 70]. Thus, while prior methods can extract 3D information using high-resolution cascaded radars [10], 3D imaging from single-chip radars generally requires additional information such as structured motion or multiple views, for example segmentation and maps using a rotating single-chip radar [24], a 3D occupancy map using multiple views [18], or high resolution images from fixed trajectories using Synthetic Aperture Radar [14, 35, 66, 67]. Instead, we show that by leveraging a sufficiently large dataset, even single frames are sufficient to recover dense angular resolution in both azimuth and elevation.

3. Data Collection System and Dataset

Dataset scale is key to training and evaluating potential foundational models. As such, we developed a scalable data collection system (Fig. 3), which we used to collect a large raw mmWave radar dataset, consisting of 1M radar-lidar-camera samples over 29 hours (Table 1). For additional details on our dataset and data collection rig, see App. A.

3.1. Data Collection System

Our data collection system, `red-rover`, was built around a TI AWR1843 Radar, Lidar, Camera, and IMU which can be easily operated via a simple web app on a mobile phone. Our system records all data to a single hot-swappable external drive via a single linux computer which handles time synchronization, minimizing turnaround time.

Table 2. **Key specifications for each setting.** Settings have varying max Doppler D_{max} and range R_{max}; all traces used a fixed resolution of 64 Doppler and 256 range bins.

Setting	Size	Length	Average Speed	D_{max}	R_{max}
`indoor`	310k	8.9h	1.0m/s	1.2m/s	11.2m
`outdoor`	372k	10.7h	1.4m/s	1.8m/s	22.4m
`bike`	333k	9.3h	5.4m/s	8.0m/s	22.4m

Table 3. **Transformer sizes.** *Layers* indicates the number of `encoder + decoder` layers; *Speed* indicates the (batched) inference throughput of each model on a single RTX 4090.

Size	Layers	Dimension	Params	Speed
`pico`	2 + 2	256 (4 heads)	3.9M	750 fps
`tiny`	3 + 3	384 (6 heads)	12.7M	320 fps
`small`	4 + 4	512 (8 heads)	28.9M	170 fps
`medium`	6 + 6	640 (10 heads)	69.4M	84 fps
`large`	9 + 9	768 (12 heads)	149M	44 fps

We also designed our data collection system to have a compact, battery-operated form factor to allow for a variety of collection modalities, including handheld and bicycle mounted. This also allows us to collect data relevant for tasks such as indoor sensing and localization, which are underrepresented in existing datasets, while still collecting automotive-like data by mounting our system to an E-bike.

3.2. Collected Data

We collected three roughly equally sized splits (Table 2) from indoor handheld, outdoor handheld, and bike-mounted settings on the CMU campus and Pittsburgh area:

- `indoor`: inside buildings at a slow to moderate walking pace, visiting multiple floors and areas within each.
- `outdoor`: neighborhoods ranging from single family detached to high density commercial zoning at a moderate to fast walking pace.
- `bike`: bike rides in different directions from a set starting point with a moderate biking pace.

Each setting features a mobile observer, with radar modulation parameters tuned for typical speeds. For sample data from each setting, see App. A.4.

4. Methodology

Using a transformer architecture (Sec. 4.1), we train our Generalizable Radar Transformer (GRT) for a range of different tasks (Sec. 4.2-4.3), and evaluate it on our dataset using a rigorous statistical methodology (Sec. 4.4).

4.1. Model Architecture

While many architectural refinements exist for vision transformers [30, 48], as well as for radar specifically [15, 26], we use a direct adaptation (Fig. 4) of the original Transformer [61] and Vision Transformer [12] to focus on measuring the fundamental properties of Radar transformers.

Radar Processing From the (slow time, TX, RX, fast time) I/Q stream, we perform a 4-Dimensional FFT to obtain (range, Doppler, azimuth, elevation) dense 4D radar data cubes of size (256, 64, 8, 2), which we provide to the model as two channels consisting of the amplitude and phase angle. This data cube is patched along the range and Doppler dimensions into patches of size $4 \times 2 (\times 8 \times 2)$, yielding an initial set of $64 \times 32 = 2048$ patch tokens.

Transformer Architecture Crucially, unlike a vision transformer [12], radar models take inputs that have different *spatial axes* than their outputs, with vastly different relative resolutions where they overlap. As such, we use a standard transformer *with a decoder* [61], with varying layers and widths (Table 3); for a full specification of our transformer architecture and training procedure, see App. B.1.

Decoder Query To handle the "change of basis" between the input and output space, we use an architecture based on Perceiver I/O [20]. We start by concatenating a (learned) output token to the encoder (similar to standard vision transformer [8]). The encoder output corresponding to the output token is then tiled into the desired decoder shape with a 3D sinusoidal positional encoding applied and is used as the input to the decoder, which attends to the encoder outputs.

4.2. Base Task: 3D Occupancy Classification

An ideal foundational model training task should be easy to gather data for (e.g., using self-supervised learning) and closely aligned with a wide range of potential downstream tasks. As such, since we are primarily concerned with understanding the *spatial* relationship between 4D radar data and 3D space, we use 3D (polar) occupancy classification – predicting the occupancy of $64 \times 128 \times 64$ range-azimuth-elevation cells – as a base task, with Lidar as a ground truth. Our task uses a binary cross entropy objective, with some weighting to correct for cell sizes; see App. B.3 for details.

Notably, in addition to being fully self-supervised, this task covers all three possible *output dimensions* (range, azimuth, and elevation), meaning that downstream tasks such as range-azimuth classification or azimuth-elevation segmentation can be cast as 2D slices of this 3D output. This enables us to fine-tune for tasks, even if they have different spatial dimensions, simply by replacing the output head and modifying the output positional encoding queries.

4.3. Other Tasks

In order to evaluate GRT's suitability as a *foundational model* for downstream fine-tuning, we use three additional tasks, each representing different output dimensions:

- **Bird's Eye View (BEV) Occupancy**: Similarly to [15, 45], we classify the 256×1024 **range-azimuth** polar occupancy using Lidar as a ground truth, with the range normalized to the radar's range resolution.

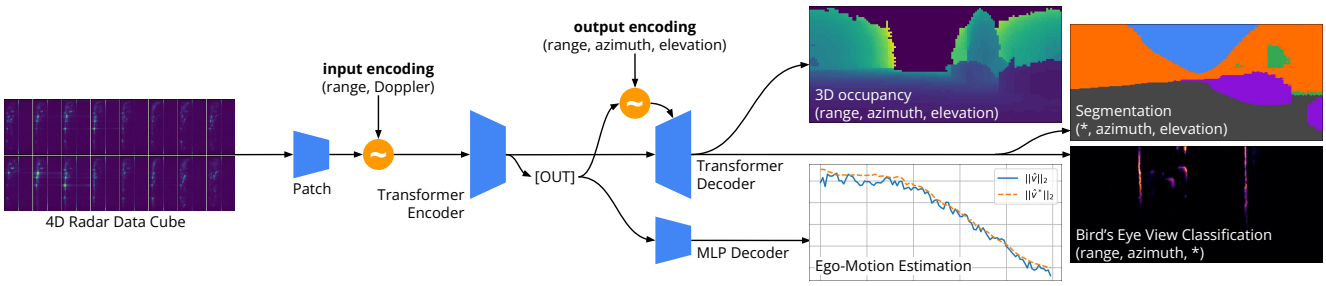


Figure 4. **The GRT architecture.** 4D radar cubes are patched with a linear projection, and a sinusoidal positional encoding is added. A transformer architecture is then used, with a transformer decoder for dense outputs and a MLP decoder for Ego-Motion estimation; different output encodings are used depending on the output axes and resolution.

- **Semantic Segmentation**: Similar to [24], we train our GRT to output 640×640 **azimuth-elevation** class labels. Since radars cannot feasibly identify many classes (e.g., poster vs. sign vs. wall) which a camera could, we use eight coarse categories: person, sky, vehicle, flat, nature, structure, ceiling, and object.
- **Ego-Motion Estimation**: Since radars can "natively" measure velocity[2], we predict the velocity of the radar relative to its current orientation. Since ego-motion estimation does not require a dense output, we replace the transformer decoder with a multi-layer-perceptron decoder head with 3 layers of 512 units.

For more detail about each task, see App. B.3.

4.4. Evaluation

Due to the cost of scaling foundational models, false positives can result in significant wasted resources, especially if associated with a costly methodological change. As such, since our dataset cannot be treated as having an "infinite sample size", we calculate upper-bounded uncertainty estimates wherever possible. In order to ensure these results are statistically accurate, we take the following steps:

- **Geo-Split**: Within each setting, ≈1.5 hours of data was reserved as a test set, which we ensured to be geographically disjoint from the training set to prevent leakage [28].
- **Sample Size correction**: Time series signals – e.g. radar-lidar-video tuples – cannot be viewed as independent samples; as such, the *effective sample size*, which we obtain from an autocorrelation-based estimate [51], must be used when calculating the standard error.
- **Paired z-Test**: Using the fact that models are evaluated on the same test traces, we use a paired z-test on the relative performance of each model with respect to a baseline.

We report each metric relative to its specified baseline by default, along with error bars for a two-sided 95% confidence interval; using our procedure and dataset, we can measure differences of 1-2% (App. B.4).

[2]Sensors which provide "absolute" pose, e.g. Lidars and Cameras, must differentiate, while IMUs must integrate.

Validation Split and Data Size Sampling. We used the last 10% of each training trace for validation (separate from the test set), with the first 90% used for training. When training on reduced dataset sizes, we use the first 9%, 18%, and 45% of each trace for training for 10%, 20%, 50% dataset sizes respectively; to reduce the variance of our experiments, we always 10% of each trace for validation.

5. Results

Using our dataset, we first ran extensive ablation (Sec. 5.1), scaling (Sec. 5.2), and fine-tuning (Sec. 5.3) experiments which show the efficacy, scalability, and generalizability of GRT. Our experiments took 874 RTX 4090-hours[3] of training time, with the `GRT-small` model taking 22 RTX 4090-hours to train.

Model Performance Despite the low resolution (only 3 TX × 4 TX antenna) of our radar, GRT is able to predict a range of outputs with remarkable quality (Fig. 5); we show additional examples in App. C.1. We also evaluated common metrics for each objective as an absolute reference (App. C.2); GRT achieves a 3D chamfer distance of 4.9 range bins, corresponding to 0.66m `indoors`, 1.6m `outdoors`, and 1.5m on `bike`.

5.1. Ablation Studies

Using our dataset and GRT, we performed ablation studies on several parameters that are independent of the underlying architecture and the task (Table 4). In particular, we find that several common practices – omitting Doppler information, using Angle-of-Arrival Estimates, and applying CFAR thresholding – result in degraded performance equivalent to more than a 10× reduction in training data.

Input Representation Our models use complex 4D radar cubes that losslessly capture all information measured by a radar; however, the common practice in radar models and data sets is to use processed higher-level representations. We benchmark three common approaches:

[3]Our experiments were run on a range of different machines with varying compute capacity, which we normalize with respect to a single RTX 4090. We only tracked training and validation time, with testing excluded.

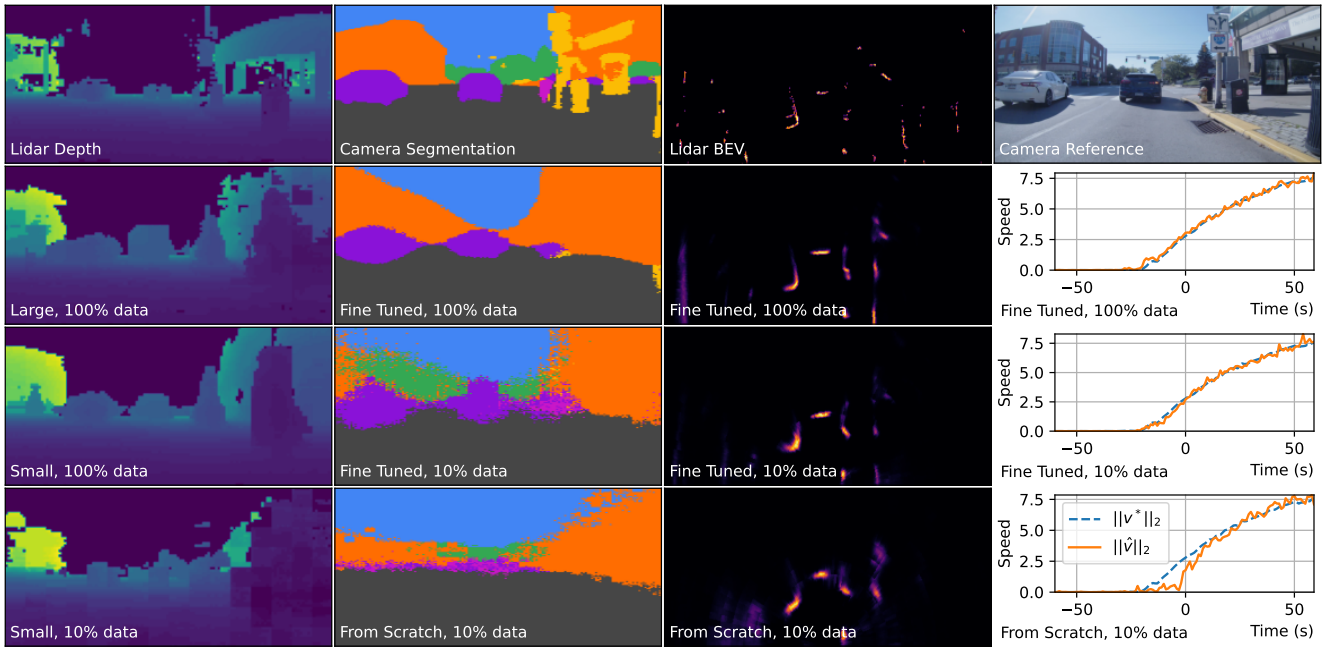


Figure 5. **Training with more data and fine tuning instead of starting from scratch lead to much higher performance.** 3D occupancy maps are less noisy as seen in rendered depth images (left), semantic segmentation is cleaner (center left), bird's eye view occupancy is sharper (center right), and velocity estimation is more accurate (right). However, scaling model size (left) does not have a large impact.

- **Real (amplitude-only) 4D data cubes** are not significantly different from complex data, indicating that the "leftover" phase from a Doppler FFT carries little additional information. Since using complex data has negligible compute overhead, we default to complex inputs.
- **Angle of Arrival Estimates** can be used to replace dense antenna measurements. Since our radar has only 2 elevation bins, we reduce the azimuth axis (8 bins) into an AoA estimate. This discards a substantial amount of information, leading to a 28.0% loss increase, equivalent to more than a 10× decrease in the size of the dataset.
- **Constant False Alarm Rate (CFAR)** processing removes weak or noisy reflectors based on local estimates of background noise [36]; this ablation uses a p-value threshold of 0.05, and zeros out rejected points. This leads to an even higher 31.5% loss increase.

Table 4. **Test loss for each ablation** (smaller is better) relative to `GRT-small` trained on our full dataset, along with 95% confidence intervals for the relative differences.

	Ablation	Relative Test Loss
Inputs	Amplitude Only	$+0.04 \pm 0.85\%$
	Angle of Arrival	$+28.0 \pm 2.12\%$
	CFAR Thresholding	$+31.5 \pm 2.38\%$
Doppler	Without Doppler FFT	$+22.5 \pm 1.76\%$
	Slow Time Shuffled	$+23.10 \pm 1.94\%$
Post-Patch Axes	Doppler-Az-El	$+6.22 \pm 1.11\%$
	Range-Az-El	$+6.27 \pm 1.10\%$
	Range-Doppler-Az-El	$+4.18 \pm 1.09\%$
Augmentations	None	$+5.87 \pm 1.32\%$
	Scale, Phase, and Flip Only	$+3.89 \pm 1.19\%$
Separate Models	Indoor Data Only	$+5.76 \pm 1.85\%$
	Outdoor Data Only	$+5.58 \pm 1.26\%$
	Bike Data Only	$+2.77 \pm 1.41\%$

Impact of Doppler With limited angular resolution, GRT is highly dependent on Doppler information, which can capture higher resolution geometry [18]. We find that removing the Doppler FFT from our processing pipeline (i.e. treating each 4D radar cube as a time series of 64 3D frames [45]) leads to a 22.5% loss increase. As an additional ablation, we also shuffle the slow-time axis to fully destroy any Doppler information; this does not lead to a futher significant loss increase, suggesting that off-the-shelf transformers cannot easily learn FFTs. Finally, we observe worse performance at low speeds since less Doppler information is available at slow speeds (App. C.3).

Patch Axes Since 4D range-Doppler-azimuth-elevation radar data cubes have four axes with different properties, they do not have an obvious counterpart to the square patches used in Vision Transformers. Benchmarking four alternatives (App. B.1), with each resulting in 2048 total patches, we find that Range-Doppler patching where the azimuth and elevation axes are "patched out" (similar to [15, 49]) is the most effective, performing ≈5% better.

Data Augmentations We develop a range of data augmentations that together provide a modest but significant performance improvement ($5.87 \pm 1.32\%$; Table 4); we pro-

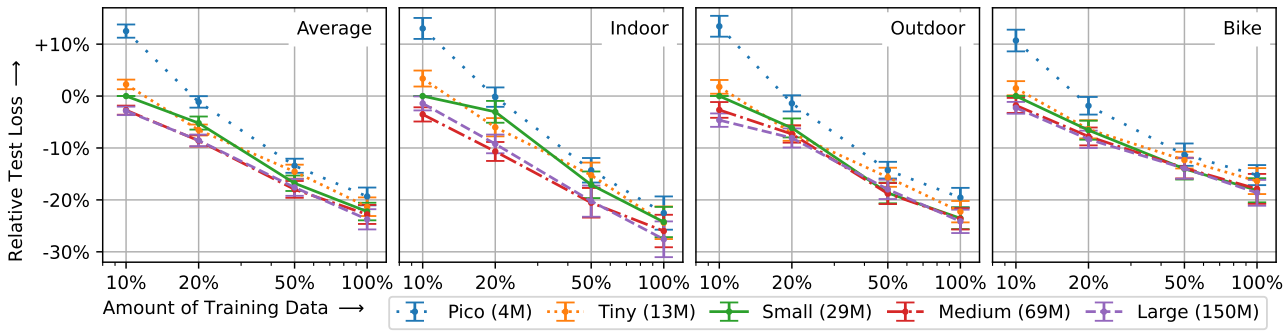


Figure 6. **Scaling laws for mmWave radar transformers** across `indoor`, `outdoor`, and `bike` test splits. Models and confidence intervals are measured relative to the `small` transformer trained on 10% of the dataset. While our models show weak scaling over model size when trained on our dataset, we see strong log-linear scaling across dataset size of ≈20% loss decrease per 10× increase in data size. For variants of this graph with respect to absolute metrics, see App. C.2.

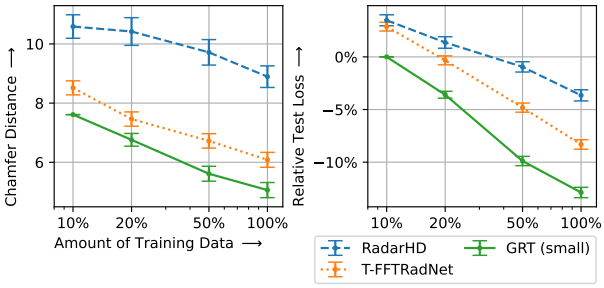


Figure 7. **GRT compared to Baselines**, measured with respect to Chamfer distance (in range bins) and test loss across different training set sizes, averaged across our dataset.

vide details about each augmentation in App. B.2.

5.2. Towards a Radar Foundational Model

Scalability vs. Baselines Since prior work on learning for single-chip radar focuses on 2D outputs, we benchmark our transformer-based approach against two prior architectures for 2D BEV Occupancy prediction: a U-Net-based model (RadarHD [45]), and a Swin Transformer-based model (T-FFTRadNet [15]), with minor architecture modifications to conform to our data dimensions (App. B.5). We find that GRT-`small` outperforms both baselines at all training splits (Fig. 7), demonstrating the suitability of transformer architectures for scalability.

Scaling Laws Training 20 different models for 5 different sizes (Table 3) and dataset sizes ranging from 10-100% of our data (Fig. 6), we observe a logarithmic improvement with data size of approximately 20% improvement per 10× increase in data, similar to early observations in computer vision [57]. This can also be seen qualitatively (Fig. 5), where models trained on more data produce much higher quality predictions. Similarly to vision transformers [57, 68], we also observe that larger models are more data efficient, although the magnitude of difference that we observe is much smaller due to our limited dataset size.

Table 5. **Chamfer Distance (in meters) for 2D BEV occupancy prediction on the Coloradar dataset [23] by location**; the geometric mean is listed to account for the varying difficulty of each location. A fine-tuned GRT model outperforms baselines trained only on Coloradar, including a state of the art diffusion-based model [70] and a U-Net based model [45].

Trace	GRT (Ours)	Diffusion [70]	RadarHD [45]
Geometric Mean	**0.98**	1.19	1.73
ARPG Lab	**0.78**	0.96	1.73
EC Hallways	**1.04**	**1.04**	1.69
Aspen	0.61	**0.51**	0.91
Longboard	**2.63**	5.47	5.40
Outdoors	**1.84**	2.37	3.10
Edgar	**0.36**	0.44	0.60

Generalizability across different settings We evaluate the ability of GRT to generalize across different settings by comparing a baseline model trained on combined `indoor`, `outdoor`, and `bike` data with models trained on each setting separately (Table 4). Despite the differences in these settings, the jointly trained model performs significantly better than models trained on each setting separately, confirming that data from different settings can be combined to train a single, stronger model.

5.3. As a Base Model for Downstream Tasks

Dataset Fine-tuning We fine-tuned a `small` GRT model on the Coloradar [23] dataset using a BEV Occupancy objective, and benchmarked the resulting model against two prior approaches trained only on ColoRadar, including a state-of-the-art diffusion model, using the same data splits and evaluation procedure [70]. Notably, despite using a modulation and resolution (128 range × 128 Doppler) which is not present in our dataset, GRT can be run without any architectural modifications, such as modifying the number of upsampling stages, as would be required for a convolutional architecture (App. B.5). After fine-tuning until vali-

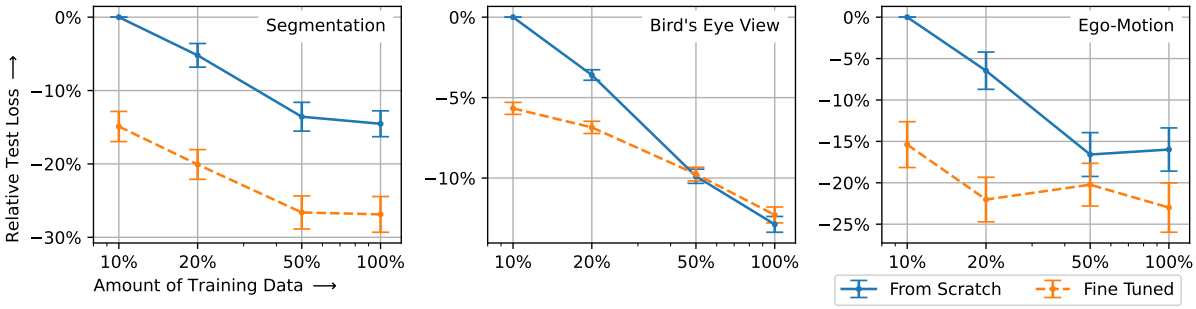


Figure 8. **Fine tuning mmWave radar transformers to different downstream tasks.** Models and confidence intervals are measured relative to the `small` transformer trained (from scratch) on 10% of the dataset. Pre-training and fine tuning strongly impacts data efficiency, equivalent to up to a $5\times$ increase in dataset size (observed in the Segmentation task).

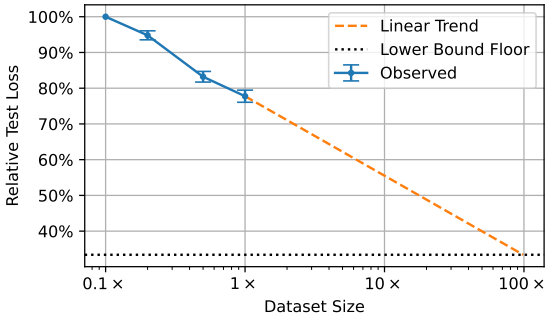


Figure 9. A linear projection of the observed logarithmic scaling to a test loss lower bound suggests that logarithmic scaling cannot continue beyond $100\times$ our current dataset

dation loss convergece ($\approx$30 minutes of training using a single RTX 4090), GRT achieves substantially lower Chamfer distance than baselines trained only on ColoRadar, showing the value of easily tunable foundational models (Table 5).

Task Fine-tuning We also fine-tuned `GRT-small` for each of our secondary tasks using 10-100% of our dataset, and compared the results with models trained from scratch on the same proportions of the dataset. Following this procedure, we find substantial performance gains equivalent to up to a $5\times$ increase in dataset size compared to training from scratch (Fig. 8). This effect is especially pronounced when less data is available, with the performance benefits of fine tuning disappearing as the dataset is scaled for the BEV Occupancy objective but staying more or less constant for the Semantic Segmentation objective. We also observe this effect as a clear qualitative difference: fine-tuned models produce sharper and more accurate predictions than their counterparts trained from scratch (Fig. 5).

5.4. How Much More Data is Needed?

Although we cannot directly observe performance saturation, we project how much data would be required to saturate a Radar Transformer using two different methods to arrive at a best guess of approximately 100M samples – $100\times$ our current dataset.

Linear Projection of Scaling Laws To lower-bound the possible test loss in our dataset, we trained a `small` model on the test set to approximate convergence. Assuming that the rate of improvement in model performance with increased training data cannot decrease, we extend our observed (Fig. 6) logarithmic scaling law to this lower bound to, in turn, estimate an upper bound for when the logarithmic trend will no longer hold (Fig. 9). This yields an estimate of $100\times$ our current dataset size. For additional details justifying our estimation of this bound, see App. C.4.

Validation Curve Trends We observe that GRTs tend to stop improving (with respect to validation loss) after $\approx$10 training epochs, regardless of model or dataset size (App. C.4); this is similar to trends observed in the training of data-constrained LLMs, which are also observed to saturate around 10 epochs [37]. Using vision transformers, whose scaling laws are well studied [68] due to the availability of internet data, as a reference, we expect training saturation to occur around $10^2 - 10^4$M samples seen. Since each epoch in our dataset corresponds to $\approx$1M samples seen, this implies that $10\times$ to $1000\times$ our current dataset size is required to delay overfitting beyond this point.

6. Conclusion

In this paper, we train a Generalizable Radar Transformer using a large, 29 hour (1M sample) dataset collected using our open-source data collection system and demonstrate that our Radar Transformer can generalize across datasets and settings, can be readily fine-tuned, and exhibits logarithmic scaling. While we believe that substantial gains are still possible through further data scaling, we hope that our dataset and baseline models will enable the community to revisit previous methods in new context and explore new capabilities made possible by a much larger dataset.

References

[1] Karim Armanious, Maurice Quach, Michael Ulrich, Timo Winterling, Johannes Friesen, Sascha Braun, Daniel Jenet, Yuri Feldman, Eitan Kosman, Philipp Rapp, et al. Bosch street dataset: A multi-modal dataset with imaging radar for automated driving. *arXiv preprint arXiv:2407.12803*, 2024. 3

[2] Jie Bai, Lianqing Zheng, Sen Li, Bin Tan, Sihan Chen, and Libo Huang. Radar transformer: An object classification network based on 4d mmw imaging radar. *Sensors*, 21(11): 3854, 2021. 3

[3] Dan Barnes, Matthew Gadd, Paul Murcutt, Paul Newman, and Ingmar Posner. The oxford radar robotcar dataset: A radar extension to the oxford robotcar dataset. In *2020 IEEE international conference on robotics and automation (ICRA)*, pages 6433–6438. IEEE, 2020. 3

[4] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021. 2

[5] Keenan Burnett, David J Yoon, Yuchen Wu, Andrew Z Li, Haowei Zhang, Shichen Lu, Jingxing Qian, Wei-Kang Tseng, Andrew Lambert, Keith YK Leung, et al. Boreas: A multi-season autonomous driving dataset. *The International Journal of Robotics Research*, 42(1-2):33–42, 2023. 3

[6] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. 3

[7] Han Cui, Shu Zhong, Jiacheng Wu, Zichao Shen, Naim Dahnoun, and Yiren Zhao. Milipoint: A point cloud dataset for mmwave radar. *Advances in Neural Information Processing Systems*, 36, 2024. 3

[8] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. *arXiv preprint arXiv:2309.16588*, 2023. 4

[9] Colin Decourt, Rufin VanRullen, Didier Salle, and Thomas Oberlin. Darod: A deep automotive radar object detector on range-doppler maps. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, pages 112–118. IEEE, 2022. 3

[10] Fangqiang Ding, Xiangyu Wen, Yunzhou Zhu, Yiming Li, and Chris Xiaoxuan Lu. Radarocc: Robust 3d occupancy prediction with 4d imaging radar. *Advances in Neural Information Processing Systems*, 37:101589–101617, 2025. 3

[11] Christopher Doer and Gert F Trommer. Yaw aided radar inertial odometry using manhattan world assumptions. In *2021 28th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS)*, pages 1–9. IEEE, 2021. 2

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 3, 4

[13] Xiangyu Gao, Guanbin Xing, Sumit Roy, and Hui Liu. Ramp-cnn: A novel neural network for enhanced automotive radar object recognition. *IEEE Sensors Journal*, 21(4): 5119–5132, 2020. 3

[14] Xiangyu Gao, Sumit Roy, and Guanbin Xing. Mimo-sar: A hierarchical high-resolution imaging algorithm for mmwave fmcw radar in autonomous driving. *IEEE Transactions on Vehicular Technology*, 70(8):7322–7334, 2021. 3

[15] James Giroux, Martin Bouchard, and Robert Laganiere. T-fftradnet: Object detection with swin vision transformers from raw adc radar signals. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4030–4039, 2023. 3, 4, 6, 7, 8

[16] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 6

[17] Wolfgang Hess, Damon Kohler, Holger Rapp, and Daniel Andor. Real-time loop closure in 2d lidar slam. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1271–1278. IEEE, 2016. 7

[18] Tianshu Huang, John Miller, Akarsh Prabhakara, Tao Jin, Tarana Laroia, Zico Kolter, and Anthony Rowe. Dart: Implicit doppler tomography for radar novel view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24118–24129, 2024. 2, 3, 6, 1, 8

[19] Cesar Iovescu and Sandeep Rao. The fundamentals of millimeter wave sensors. *Texas Instruments*, pages 1–8, 2017. 3

[20] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. Perceiver io: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*, 2021. 4

[21] Yi Jin, Marcel Hoffmann, Anastasios Deligiannis, Juan-Carlos Fuentes-Michel, and Martin Vossiek. Semantic segmentation-based occupancy grid map learning with automotive radar raw data. *IEEE Transactions on Intelligent Vehicles*, 9(1):216–230, 2023. 3

[22] Nidhi Kalra and Susan M Paddock. Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? *Transportation Research Part A: Policy and Practice*, 94:182–193, 2016. 3

[23] Andrew Kramer, Kyle Harlow, Christopher Williams, and Christoffer Heckman. Coloradar: The direct 3d millimeter wave radar dataset. *The International Journal of Robotics Research*, 41(4):351–360, 2022. 2, 3, 7

[24] Haowen Lai, Gaoxiang Luo, Yifei Liu, and Mingmin Zhao. Enabling visual recognition at radio frequency. In *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*, pages 388–403, 2024. 2, 3, 5, 9

[25] Xinrong Li, Xiaodong Wang, Qing Yang, and Song Fu. Signal processing for tdm mimo fmcw millimeter-wave radar sensors. *IEEE Access*, 9:167959–167971, 2021. 2

[26] Yu-Jhe Li, Shawn Hunt, Jinhyung Park, Matthew O'Toole, and Kris Kitani. Azimuth super-resolution for fmcw radar in

autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17504–17513, 2023. 3, 4

[27] Jaime Lien, Nicholas Gillian, M Emre Karagozler, Patrick Amihood, Carsten Schwesig, Erik Olson, Hakim Raja, and Ivan Poupyrev. Soli: Ubiquitous gesture sensing with millimeter wave radar. *ACM Transactions on Graphics (TOG)*, 35(4):1–19, 2016. 1

[28] Adam Lilja, Junsheng Fu, Erik Stenborg, and Lars Hammarstrand. Localization is all you evaluate: Data leakage in online mapping datasets and how to fix it. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22150–22159, 2024. 5

[29] Teck-Yian Lim, Spencer A Markowitz, and Minh N Do. Radical: A synchronized fmcw radar, depth, imu and rgb camera data dataset with low-level fmcw radar signals. *IEEE Journal of Selected Topics in Signal Processing*, 15(4):941–953, 2021. 3, 1

[30] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 4

[31] Ilya Loshchilov, Frank Hutter, et al. Fixing weight decay regularization in adam. *arXiv preprint arXiv:1711.05101*, 5, 2017. 6

[32] Chris Xiaoxuan Lu, Muhamad Risqi U Saputra, Peijun Zhao, Yasin Almalioglu, Pedro PB De Gusmao, Changhao Chen, Ke Sun, Niki Trigoni, and Andrew Markham. milliego: single-chip mmwave radar aided egomotion estimation via deep sensor fusion. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, pages 109–122, 2020. 3

[33] Sohrab Madani, Jayden Guan, Waleed Ahmed, Saurabh Gupta, and Haitham Hassanieh. Radatron: Accurate detection using multi-resolution cascaded mimo radar. In *European Conference on Computer Vision*, pages 160–178. Springer, 2022. 3

[34] Bence Major, Daniel Fontijne, Amin Ansari, Ravi Teja Sukhavasi, Radhika Gowaikar, Michael Hamilton, Sean Lee, Slawomir Grzechnik, and Sundar Subramanian. Vehicle detection with automotive radar using deep learning on range-azimuth-doppler tensors. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019. 3

[35] Babak Mamandipoor, Greg Malysa, Amin Arbabian, Upamanyu Madhow, and Karam Noujeim. 60 ghz synthetic aperture radar for short-range imaging: Theory and experiments. In *2014 48th Asilomar Conference on Signals, Systems and Computers*, pages 553–558. IEEE, 2014. 3

[36] Gary Minkler and Jing Minkler. Cfar: the principles of automatic radar detection in clutter. *Nasa sti/recon technical report a*, 90:23371, 1990. 2, 3, 6

[37] Niklas Muennighoff, Alexander Rush, Boaz Barak, Teven Le Scao, Nouamane Tazi, Aleksandra Piktus, Sampo Pyysalo, Thomas Wolf, and Colin A Raffel. Scaling data-constrained language models. *Advances in Neural Information Processing Systems*, 36:50358–50376, 2023. 8

[38] Farzan Erlik Nowruzi, Dhanvin Kolhatkar, Prince Kapoor, Fahed Al Hassanat, Elnaz Jahani Heravi, Robert Laganiere, Julien Rebut, and Waqas Malik. Deep open space segmentation using automotive radar. In *2020 IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM)*, pages 1–4. IEEE, 2020. 3

[39] Itai Orr, Moshik Cohen, and Zeev Zalevsky. High-resolution radar road segmentation using weakly supervised learning. *Nature Machine Intelligence*, 3(3):239–246, 2021. 3

[40] Arthur Ouaknine, Alasdair Newson, Patrick Pérez, Florence Tupin, and Julien Rebut. Multi-view radar semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15671–15680, 2021. 3

[41] Arthur Ouaknine, Alasdair Newson, Julien Rebut, Florence Tupin, and Patrick Pérez. Carrada dataset: Camera and automotive radar with range-angle-doppler annotations. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 5068–5075. IEEE, 2021. 3

[42] Dong-Hee Paek, Seung-Hyun Kong, and Kevin Tirta Wijaya. K-radar: 4d radar object detection for autonomous driving in various weather conditions. *Advances in Neural Information Processing Systems*, 35:3819–3829, 2022. 3

[43] Andras Palffy, Jiaao Dong, Julian FP Kooij, and Dariu M Gavrila. Cnn based road user detection using the 3d radar cube. *IEEE Robotics and Automation Letters*, 5(2):1263–1270, 2020. 3

[44] Akarsh Prabhakara, Diana Zhang, Chao Li, Sirajum Munir, Aswin C Sankaranarayanan, Anthony Rowe, and Swarun Kumar. Exploring mmwave radar and camera fusion for high-resolution and long-range depth imaging. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3995–4002. IEEE, 2022. 3

[45] Akarsh Prabhakara, Tao Jin, Arnav Das, Gantavya Bhatt, Lilly Kumari, Elahe Soltanaghai, Jeff Bilmes, Swarun Kumar, and Anthony Rowe. High resolution point clouds from mmwave radar. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4135–4142. IEEE, 2023. 2, 3, 4, 6, 7, 1, 8

[46] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 3

[47] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 13

[48] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12179–12188, 2021. 4

[49] Julien Rebut, Arthur Ouaknine, Waqas Malik, and Patrick Pérez. Raw high-definition radar for multi-task learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17021–17030, 2022. 3, 6

[50] Mark A Richards et al. *Fundamentals of radar signal processing*. Mcgraw-hill New York, 2005. 2, 3

[51] Christian P Robert, George Casella, and George Casella. *Monte Carlo statistical methods*. Springer, 1999. 5, 8

[52] Hermann Rohling. Radar cfar thresholding in clutter and multiple target situations. *IEEE Transactions on Aerospace and Electronic Systems*, AES-19(4):608–621, 1983. 2, 3

[53] Mark E Russell, Arthur Crain, Anthony Curran, Richard A Campbell, Clifford A Drubin, and Willian F Miccioli. Millimeter-wave radar sensor for automotive intelligent cruise control (icc). *IEEE Transactions on microwave theory and techniques*, 45(12):2444–2453, 1997. 2

[54] Marcel Sheeny, Emanuele De Pellegrin, Saptarshi Mukherjee, Alireza Ahrabian, Sen Wang, and Andrew Wallace. Radiate: A radar dataset for automotive perception in bad weather. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–7. IEEE, 2021. 3

[55] Xian Shuai, Yulin Shen, Yi Tang, Shuyao Shi, Luping Ji, and Guoliang Xing. millieye: A lightweight mmwave radar and camera fusion system for robust object detection. In *Proceedings of the International Conference on Internet-of-Things Design and Implementation*, pages 145–157, 2021. 3

[56] Akash Deep Singh, Sandeep Singh Sandha, Luis Garcia, and Mani Srivastava. Radhar: Human activity recognition from point clouds generated through a millimeter-wave radar. In *Proceedings of the 3rd ACM Workshop on Millimeter-wave Networks and Sensing Systems*, pages 51–56, 2019. 3

[57] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017. 7

[58] Texas Instruments. Automated doors reference design using ti mmwave sensors. Design Guide TIDEP-01018, Texas Instruments, Dallas, TX, 2018. 1

[59] *AWR1843AOP Single-chip 77- and 79-GHz FMCW mmWave Sensor Antennas-OnPackage (AOP)*. Texas Instruments, 2021. 2, 3

[60] Claudia Vasanelli, Fabian Roos, Andre Durr, Johannes Schlichenmaier, Philipp Hugler, Benedikt Meinecke, Maximilian Steiner, and Christian Waldschmidt. Calibration and direction-of-arrival estimation of millimeter-wave radars: A practical introduction. *IEEE Antennas and Propagation Magazine*, 62(6):34–45, 2020. 2

[61] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017. 3, 4, 6

[62] Christian Waldschmidt, Juergen Hasch, and Wolfgang Menzel. Automotive radar—from first efforts to future systems. *IEEE Journal of Microwaves*, 1(1):135–148, 2021. 2

[63] Yizhou Wang, Zhongyu Jiang, Xiangyu Gao, Jenq-Neng Hwang, Guanbin Xing, and Hui Liu. Rodnet: Radar object detection using cross-modal supervision. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 504–513, 2021. 3

[64] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in neural information processing systems*, 34: 12077–12090, 2021. 7

[65] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, pages 10524–10533. PMLR, 2020. 6

[66] Hiroyoshi Yamada, Takumi Kobayashi, Yoshio Yamaguchi, and Yuuichi Sugiyama. High-resolution 2d sar imaging by the millimeter-wave automobile radar. In *2017 IEEE Conference on Antenna Measurements & Applications (CAMA)*, pages 149–150. IEEE, 2017. 3

[67] Muhammet Emin Yanik and Murat Torlak. Near-field mimo-sar millimeter-wave imaging with sparsely sampled aperture data. *Ieee Access*, 7:31801–31819, 2019. 3

[68] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12104–12113, 2022. 3, 7, 8, 13

[69] Ao Zhang, Farzan Erlik Nowruzi, and Robert Laganiere. Raddet: Range-azimuth-doppler based radar object detection for dynamic road users. In *2021 18th Conference on Robots and Vision (CRV)*, pages 95–102. IEEE, 2021. 2, 3

[70] Ruibin Zhang, Donglai Xue, Yuhan Wang, Ruixu Geng, and Fei Gao. Towards dense and accurate radar perception via efficient cross-modal diffusion model. *IEEE Robotics and Automation Letters*, 2024. 3, 7

[71] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 633–641, 2017. 7

Towards Foundational Models for Single-Chip Radar

Supplementary Material

A. Data Collection and Dataset Details

Our data collection system, `red-rover`[4], is fully open source. The data collection rig and control app are shown in detail in Fig. 11, along with the system in its bike-mounted configuration in Fig. 10; all parts used can be purchased either off-the-shelf or 3D-printed using an ordinary 3D printer.

Bill of Materials The total cost of the bill of materials for our data collection system is $4,440 for the base system (Table 6). Note that the Lidar is the primary contributor to the cost of our data collection rig; while we use an OS0-128 ($12,000), it can be substituted for an OS0-64 ($8,000) or OS0-32 ($4,000) without any hardware or software modifications, though at the cost of reduced Lidar data quality.

A detailed bill of materials including all parts used in the data collection rig (along with CAD files for 3D printed parts) is available in our `red-rover` project repository.

Resource Usage For the configurations which we used to collect I/Q-1M, our data collection rig has the following overall characteristics:

- Data rate: ≈120GB/hour (≈33MB/s — 260Mbps), with some variation depending on the compressibility of the data. In practice, we do not find storage to be a substantial limitation, with the total dataset size being ≈3.5TB.
- Power consumption: ≈80W average. Using a 240Wh AC battery bank, this results in around 3 hours of battery life.

A.1. Sensors

Our data collection rig includes a radar, lidar, camera, and IMU, and records a total data bitrate of ≈260 Mbps. Almost half of the bitrate is consumed by the radar (126 MBbps), with the remainder being split between the Camera and Lidar, with a negligible amount data recorded from the IMU.

Radar "Boost" development boards for the TI 77GHz single-chip mmWave radar family[5] are commonly used in academic research, and we are not aware of any raw single-chip radar datasets – or tooling for data collection – which uses other radars. As such, we use the AWR1843Boost Radar (and a DCA1000EVM capture card), which is commonly used in prior literature [18, 29, 45].

The AWR1843Boost has 3 transmit (TX) and 4 receive (RX) antennas, resulting in 8 azimuth and 2 elevation bins (Fig. 12). We configured our radar to record 256 range

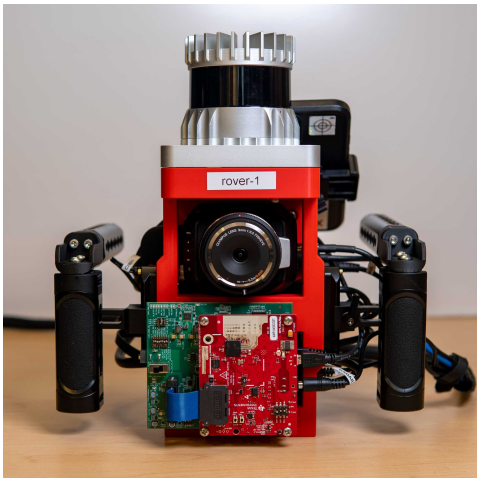
Figure 10. **Our data collection system, `red-rover`, in its bike-mounted configuration.** The sensors are mounted to a front rack, while the support electronics are mounted in the center frame and the battery at the rear for balance and stability.

Table 6. **Bill of materials and approximate cost of major components** as of time of writing, in US Dollars; carrying equipment (e.g., backpack, E-bike) and miscellaneous items under $100 (e.g. cables, screws) are not listed.

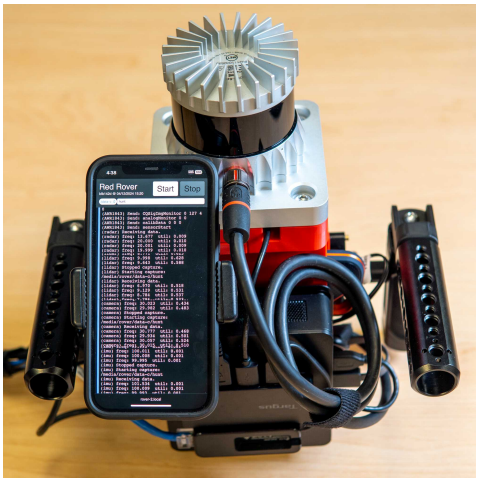
Item	Cost
Ouster OS0-32/64/128 Lidar	$4,000-$12,000
Data Collection Computer	$1,000
Black Magic Micro Studio Camera	$1,000
Magewell USB-SDI Capture Card	$300
OM Systems 9mm f/8 Fisheye Lens	$100
TI DCA1000EVM Radar Capture Card	$600
TI AWR1843Boost Radar	$300
XSens MTi-3 AHRS Development Kit	$450
External Storage Drive	$330
Battery	$240
Hardware for Handles	$120
Total	$4,440 + Lidar

[4] As the successor to our previous `rover` data collection system [18], `red-rover` is named for its distinctive red color.

[5] TI produces "Boost" series development boards across its range of 77GHz radars including the AWR1843Boost, the largely identical IWR1843Boost, and the AWR1642Boost, which is equivalent to the AWR1843Boost with its middle transmit antenna removed.



(a) Data collection rig from the front; the Lidar, Camera, and Radar (red PCB) along with its capture card (green PCB) are visible, while the IMU is hidden inside the red (3D-printed) plastic structure.



(b) Data collection rig from behind, showing the control app; the app allows users to specify metadata, then `start` and `stop` data collection. A live console displays logged messages and errors for each sensor.

Figure 11. **Close-up views of the handheld data collection rig from the front and back.**

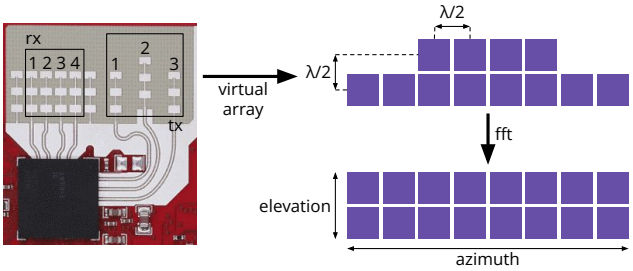


Figure 12. **Antenna configuration of the TI AWR1843Boost radar**. The 12 virtual antennas (top right) created by the radar's 3TX × 4RX antenna array (left) result in 2 elevation and 8 azimuth bins (lower right).

bins and 64 Doppler bins at 20 Hz, with varying range and Doppler resolutions depending on the setting; see Table 7 for detailed specifications. In our dataset, we also collect raw, uncompressed I/Q streams which are quantized as 16-bit integers by the radar; with the modulations used in our dataset, the radar has a total bitrate of 126 Mbps.

Crucially, we do not to use a high-resolution imaging radar (e.g., the 12 × 16 antenna TI MMWCAS-RF-EVM): in addition to their larger size, weight, and power, imaging radars have an order-of-magnitude higher raw data rate (e.g., ≈2gbps for an equivalent modulation using the TI MMWCAS-RF-EVM), which substantially increases the engineering and infrastructure cost of collecting raw data, while also making continuous live streaming and real-time deployment impractical.

Lidar We use an Ouster OS0-128 recording 2048 azimuth bins (1024 forward-facing) and 128 elevation beams at 10Hz. In practice, we find that the OS0 Lidar has a maximum range of 20-25m: while points further away can still *sometimes* be detected, objects are not consistently detected. As such, while our radar can detect objects at much further ranges, the Lidar forces us to restrict the maximum range used in our dataset; we plan to acquire a longer-range Lidar for future iterations of our dataset.

The lidar depth is LZMA-compressed, resulting in a bitrate of 14 Mbps. While we do not use these channels in our paper, we also collect the reflectance and near infrared background intensity, with a typical data rate of 9 Mbps and 22 Mbps, respectively.

Table 7. **Full radar configurations for each setting.** With a fixed frame size of $N_r = 256$ samples/chirp and $N_d = 64$ chirps/frame, configuring the radar's chirp rate, ADC sample rate, and chirp slope determines the range resolution $\Delta R = \frac{F_s c}{2N_r}$ and Doppler resolution $\Delta D = \frac{\lambda}{2N_d T_c}$ along with maximum range $R_{\max} = \frac{F_S c}{2s}$ and maximum Doppler $D_{\max} = \frac{\lambda}{4T_c}$, where λ is the radar's wavelength (77GHz; $\lambda = 3.9$mm) and c is the speed of light.

Setting	`indoor`	`outdoor`	`bike`
Chirp Time T_c	777μs	537μs	120μs
Sample Rate F_s	5MHz	5MHz	10MHz
Chirp Slope S	67MHz/μs	34MHz/μs	34MHz/μs
ΔR	4.4cm	8.7cm	8.7cm
$R_{\max}$	11.2m	22.4m	22.4m
ΔD	3.8cm/s	5.6cm/s	24.9cm/s
$D_{\max}$	1.2m/s	1.8m/s	8.0m/s

Camera We use a Black Magic Micro Studio Camera with an OM Systems 9mm f/8 Fisheye lens, recording at 1080p, 30 fps; frames are recorded as a MJPEG video, resulting in a typical bitrate of 88 Mbps. To minimize motion

Table 8. **Comparison with other mmWave radar datasets with raw data (4D data cube or equivalent), and a selection of other large datasets without raw data.** A *frame* in our table refers to the number of unique radar-sensor tuples. Our dataset is significantly larger than previous radar datasets, enabling us to scale up training.

Radar Type	Dataset	4D Data Cube	Dataset Size	Other Sensors
Single Chip	**IQ-1M (Ours)**	**Yes**	**29 hours (1M frames)**	Lidar, Camera, IMU
	MilliPoint [7]	No (3D Points)	6.3 Hours (545k frames)	Depth Camera
	nuScenes [6]	No (3D Points)	5.5 hours (400k frames)	Lidar, Camera, IMU, GPS
	RaDICal [29]	**Yes**	3.6 Hours (394k frames)	Depth Camera, IMU
	Coloradar [23]	**Yes**	2.4 hours (82k frames)	Lidar, IMU
	CRUW [63]	No (2D Map)	3 hours (400k frames)	Stereo Cameras
	RadarHD [45]	**Yes**	200k frames	Lidar
	CARRADA [41]	No (3D Cube)	21 Minutes (13k frames)	Camera
	RADDet [69]	**Yes**	10k frames	Camera
Cascaded	Radatron [33]	No (3D Cube)	4.2 hours (152k frames)	Camera
	K-radar [42]	**Yes**	35k frames	Lidar, Camera, IMU, GPS
	RADIal [49]	**Yes**	20k frames	Lidar, Camera, GPS
Mechanical	Oxford Radar RobotCar [3]	No (2D Image)	17 Hours (240k Frames)	Lidar, Camera, GPS
	RADIATE [54]	No (2D Image)	5.0 hours (72k frames)	Lidar, Camera, GPS
	Boreas [5]	No (2D Image)	350km	Lidar, Camera

blur, the camera is set to 18 db gain; the shutter speed is set to automatic. Note that while our camera and capture card are capable of 60 fps recording, we record only 30 fps since we find that 30 fps recording is far more stable than 60 fps (especially with regard to dropped frames), and since since the downstream tasks which we envision cannot easily take advantage of 60 fps video.

IMU We include a XSens MTi-3 IMU which is used for Cartographer SLAM in conjunction with our Lidar. The IMU records acceleration, angular velocity, and rotation at 100 Hz, with a total bitrate of 35 Kbps.

A.2. Time Synchronization

While each sensor is recorded against the same system clock, we asynchronously record each at its full "native" speed. To generate radar-lidar-camera samples, we align higher frequency sensors (radar – 20Hz; camera – 30Hz) to the Lidar (10Hz) by selecting the nearest sample in time to each lidar frame. Since our data collection implementations for each sensor have variable initialization and de-initialization time, we also trim regions at the start and end of each trace which do not have coverage from all sensors.

A.3. Comparison with Other Datasets

Table 8 enumerates a number of radar datasets sorted by radar type, along with their sizes and included sensors. Since different types of radars have substantially different operating modes, modulations, and data characteristics and dimensions, we focus on single-chip radars. In this category, prior datasets generally use TI single-chip radars such as the TI AWR1843 family which we use [59].

Fine-Tuning Experiment In our fine-tuning experiments, we elected to use the Coloradar [23] dataset due to its inclusion of high-quality Lidar depth data and extensive prior work using this dataset. We considered, but opted not to use, the following datasets:

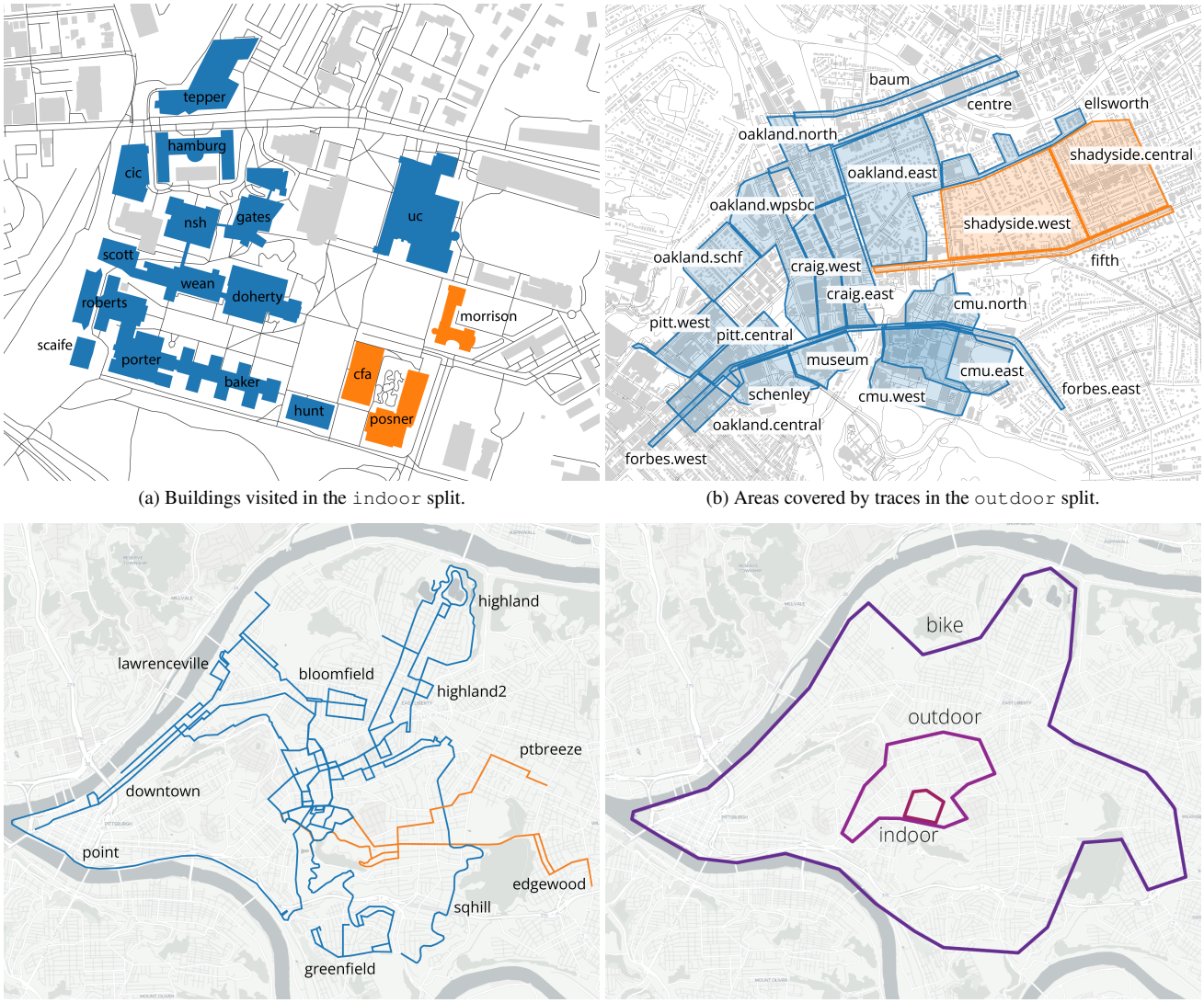
- RaDICal [29]: the depth cameras used have poor performance, especially beyond very close ranges. Likely because of this issue, we are also not aware of a substantial body of prior work using this dataset as a benchmark.
- RADDet [69]: RADDet is an extremely popular object detection dataset, and a good candidate for fine-tuning. Unfortunately, while we have been able to obtain the raw ADC data, the ground truth labels, video, and other metadata are no longer available as of time of writing.
- CRUW [63]: While the original CRUW dataset appears to include lower-level data, only 2D range-azimuth maps are publicly available.

A.4. Dataset Details

Our dataset was collected on and around the CMU campus and in the Pittsburgh area, and includes three data settings: handheld indoors, handheld outdoors, and on a bike. In addition to maps of data collection areas (Fig. 13), we also provide representative samples from each dataset (Fig. 14).

`indoor`: The indoor setting was collected from publicly accessible areas within the CMU campus. Each trace generally represents a different floor (or multiple floors, in cases where each floor is relatively small). Additionally, each floor was covered twice: once in a forward-facing configuration (with the velocity mostly aligned with the radar's orientation), and once in a "sideways facing" configuration (with the velocity mostly orthogonal to the radar).

`outdoor`: Roughly 30-minute-long traces were collected within contiguous areas with minimal overlap, with the radar generally facing forward. The areas visited include CMU and Pitt university campuses, commercial areas rang-



(a) Buildings visited in the `indoor` split.

(b) Areas covered by traces in the `outdoor` split.

(c) Traces in the `bike` split.

(d) Overview of the data collection extents for each setting.

Figure 13. **Maps of the train (blue) and test (orange) splits for each setting.**

ing from medium to high density, and residential areas ranging from single-family detached to high rise apartment buildings, as well as a variety of streets ranging from small alleys to busy "stroads."

`bike`: Data was collected on approximately 60-minute-long round-trips[6] originating from our lab; each trace is split into an inbound and outbound leg covering mostly the same path, but in different directions. Note that there is some overlap between the areas covered in the train and test splits at "bottlenecks" near the CMU campus; when viewed as a whole, we believe this is negligible.

[6]As one the authors was struck by a vehicle while collecting data on bike, we urge any efforts to replicate or extend this split to minimize mental load during data collection and use caution when planning routes. Thankfully, the author was uninjured, though the radar was destroyed.

B. Method Details

GRT uses a standard encoder-decoder transformer network (App. B.1). We also document our data augmentations (App. B.2), training tasks (App. B.3), evaluation procedure (App. B.4), and baselines (App. B.5).

B.1. GRT Training & Hyperparameters

GRT uses a standard transformer architecture with sinusoidal positional encodings, and experimentally obtained radar-specific patching parameters. For a summary of key hyperparameters and architecture parameters, see Table 9.

Architecture Unless specified otherwise, GRT's architecture uses the following common parameters:

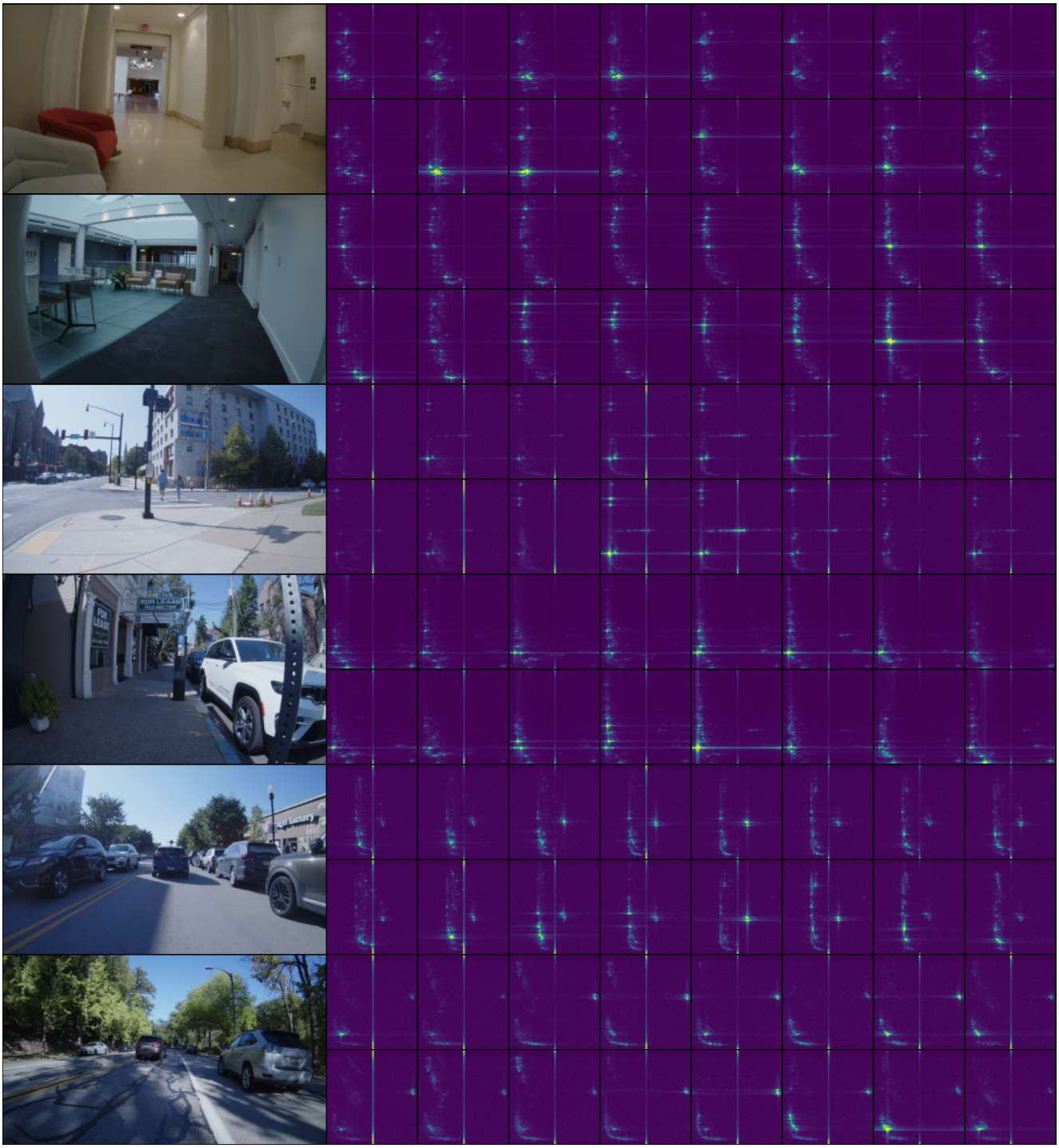


Figure 14. **Representative samples from our dataset showing camera and range-Doppler frames from the `indoor` (top), `outdoor` (middle), and `bike` (bottom) settings**. Each radar plot shows the range-Doppler image of a single (azimuth, elevation) bin. When the radar is configured with a range and Doppler resolution which is appropriate for each setting, the resulting range-Doppler frames are remarkably similar at a visual level. Note that common types of radar noise and artifacts such as a zero doppler artifact (the straight line at the center of each frame) and range-Doppler bleed (horizontal and vertical lines coming from bright reflectors) are clearly visible in these examples.

Table 9. **Key Hyperparameters for GRT.** Except for model layers and dimensionality, which we perform scaling law ablations on, these hyperparameters are taken from common transformer design practices as of time of writing.

Input Patch Size	128 $(4 \times 2 \times 8 \times 2)$
Input Number of Patches	2048
Output Number of Patches	1024
Layers	4 – 18
Model Dimension	256 – 768
Dimensions Per Head	64
Expansion Ratio	4.0
Transformer Norm	"pre-norm"
Activation	GeLU
Dropout	0.1
Batch Size	32
Optimizer	AdamW
Warmup	100 Steps
Learning Rate	10^{-4}

- We always use a simple linear patch and unpatch layers, with the appropriate output dimensionality depending on the task.
- All layers use a GeLU activation [16].
- Transformers use "pre-norm" (norm before, instead of after the transformer layer), which is generally regarded as more stable [65]; when using "post-norm" (as in the original transformer architecture [61]), we find that GRT often diverges due to numerical instability at initialization.
- Each transformer layer has a fixed expansion ratio of 4.0 and a dropout of 0.1.

Positional Encodings In both encoder and decoder positional embeddings, we use a simple N-dimensional encoding which divides the number of features equally between each dimension and independently applies a sinusoidal encoding for the coordinate in that axis. To facilitate fine-tuning for tasks with different output resolutions, we also normalize the frequencies by the total length of each axis so that different resolutions result in the same frequency range.

Training When training GRT, we use the following parameters for all models:
- We apply a range of data augmentations, which we find to provide a $\approx 5\%$ benefit (App. B.2).
- We always use a fixed batch size of 32. When training on platforms with different GPU counts, the batch is split equally between each GPU.
- All models are trained with a learning rate of 10^{-4} using the AdamW [31] optimizer with a warmup period of 100 steps. We find this warmup period to be essential in order to avoid initialization instability and NaN gradients.
- Each model was trained until the validation loss stopped decreasing, as defined by three consecutive checkpoints without a decrease in validation loss, with two checkpoints taken each epoch.

Fine-Tuning Fine-tuning uses the same procedure as for training, including termination after the validation loss stops decreasing. The model is not frozen, with all weights being trainable during the tuning process. In cases where the output dimension does not match the input dimension (e.g., 8-channel one-hot classification outputs for the Semantic Segmentation objective vs. 1-channel binary classification outputs for occupancy objectives), the output layer is also re-initialized.

Patch Size We use a patch size of 128 bins (4 range, 2 Doppler, 8 azimuth, 2 elevation) in the encoder, resulting in 2048 input patches, and square (or cubic) patches for each output sized to maintain a fixed decoder sequence length of 1024 patches. Note that this "patches out" the azimuth and elevation axes in the encoder; while we empirically determined that this leads to the best performance on our dataset (Sec. 5.1), we expect the optimal patch dimensions to vary depending on the input radar resolution.

Patch Size Alternatives In addition to the above patch size, we tested the following alternatives:
- **Range-Doppler-azimuth-elevation**: carefully selecting our patch size to keep all four axes, we create 16 range $\times$ 8 Doppler $\times$ 8 azimuth $\times$ 2 elevation patches. This results in a $4.18 \pm 1.09\%$ increase in test loss.
- **Range-azimuth-elevation**: we eliminate the Doppler axis for 128 Range $\times$ 8 azimuth $\times$ 2 elevation patches. This results in a $6.27 \pm 1.10\%$ increase in test loss.
- **Doppler-azimuth-elevation**: we eliminate the Range axis (as much as possible) to obtain 64 Doppler $\times$ 2 range $\times$ 8 azimuth $\times$ 2 elevation patches. This results in a $6.22 \pm 1.11\%$ increase in test loss.

B.2. Data Augmentations

We develop a range of data augmentations, which we ablate in two groups: *Scale, Phase, and Flip Only*, and *Full* augmentations, which we use by default.

Scale, Phase, and Flip Only This group includes "simple" augmentations which can be calculated pixel-wise:
- `radar_scale`: random scaling applied to the magnitude of the 4D radar data cube, with log-normal distribution $\exp(\mathcal{N}(0, 0.2^2))$ clipped to $[\exp(-2), \exp(2)]$.
- `radar_phase`: random phase offset of $\text{Unif}(-\pi, \pi)$ applied to the phase of the 4D radar data cube (except in ablations where no phase information is provided to the model).

- `azimuth_flip`: random flipping (with probability 0.5) along the azimuth axis, i.e., swapping left and right. Note that this augmentation also affects the ground truth for each task.
- `doppler_flip`: random flipping (with probability 0.5) along the Doppler axis, i.e. swapping positive and negative Doppler. This is equivalent to reversing the direction of travel of the sensor and all other objects in the scene; as such, this augmentation also affects the ground truth velocity by reversing the velocity vector.

Note that we do not apply an `elevation_flip` since the ground is always down!

Full In addition to the above augmentations, we include augmentations which are equivalent to random cropping:

- `range_scale`: ranges are multiplied by a $\text{Unif}(1.0, 2.0)$ scale, with the radar data cube being cropped and scaled appropriately using a bilinear interpolation. The ground truth occupancy and Bird's Eye View are also scaled accordingly.
- `speed_scale`: Doppler velocities are multiplied by a log-normal $\exp(\mathcal{N}(0, 0.2^2))$ distribution, clipped to $[\exp(-2), \exp(2)]$. All scaling is done with bilinear interpolation. If the velocity is scaled down, we zero-fill any extra bins; if velocity is scaled up, we "wrap" the Doppler velocity around to emulate the ambiguity of Doppler velocity modulo the Doppler bandwidth. Finally, the ground truth ego-motion velocity is scaled to match.

B.3. Task Details

3D Occupancy Classification Our 3D polar occupancy task uses a binary cross-entropy loss on polar grid cells which are created by the product set of the radar's range resolution with the Lidar's azimuth and elevation resolution. The loss is further scaled and balanced for the following:

- To facilitate joint training between different radar modulation, we normalize distances by the radar range resolution, resulting in a fixed output grid for each setting.
- To correct for the sparsity of 3D occupancy grids, occupied cells are weighted greater (64.0) than unoccupied cells (1.0).
- Since polar occupancy cells are larger when further away, we correct for the cell size, which is proportional to r^2.

Finally, to manage the memory required by dense 3D prediction, we apply a $4\times$ range, $8\times$ azimuth, and $2\times$ elevation decimation, resulting in (64 range $\times$ 128 azimuth $\times$ 64 elevation) bins, which we output with 1024 $(8\times8\times8)$ patches. For decimated range-azimuth-elevation grid r, θ, ϕ and 0-1 occupancy Y^*, this corresponds to the following loss:

$$\mathcal{L}(\hat{Y}_{r,\phi,\theta}, Y^*_{r,\phi,\theta}) = r^2(1.0 + 63.0Y^*_{r,\phi,\theta})\text{BCE}(\hat{Y}_{r,\phi,\theta}, Y^*_{r,\phi,\theta}) \quad (1)$$

In addition to the test loss, we compute the Chamfer distance (by treating each occupied cell as a point), and the mean absolute error of depth estimates obtained by finding the first occupied cell along the range axis of our range-azimuth-elevation occupancy.

Bird's Eye View (BEV) Occupancy We use the same binary cross-entropy and Dice loss mixture as [45], and output a 256×1024 range-azimuth polar occupancy grid which corresponds to the native range resolution of the radar and the native azimuth of the Lidar, restricted to forward-facing bins. We output 1024 (16×16) patches.

In addition to the test loss, we also compute the Chamfer distance, using the same procedure as for 3D occupancy.

Semantic Segmentation We use the 640×640 output of `segformer-b5` [64], trained on ADE20k [71], as the ground truth for this task. We aggregate the ADE20k classes into eight broad categories (arranged by index):

0. `ceiling`: any structure viewed from below; mostly seen `indoors`.
1. `flat`: flat, walkable surfaces such as sidewalks and roads. Grass and other vegetation are excluded, and included in `nature` instead.
2. `nature`: vegetation and other natural items such as grass, shrubs, trees, and water.
3. `object`: miscellaneous small objects such as furniture which are not included in the `structure` category.
4. `person`: any person who is not inside a vehicle or riding a vehicle.
5. `sky`: the sky.
6. `structure`: large man-made items such as buildings, fences, and shelters.
7. `vehicle`: cars, trucks, buses, and other vehicles.

We output 1024 $(5\times5)\times8$ patches and train using a simple binary cross-entropy loss. Finally, in addition to the test loss, we also calculate the mIOU (mean intersection over union), accuracy, and top-2 accuracy.

Ego-Motion Estimation We fuse our IMU and Lidar data Cartographer SLAM [17], which we project back to the sensor's frame of reference to obtain Ego-Motion ground truth, and manually exclude regions where SLAM failed (typically due to a lack of Lidar features, e.g. bridges or tunnels, totaling $\approx 1\%$ of our dataset).

During training, we first normalize the velocity with respect to the Doppler resolution (i.e. measuring each component in Doppler bins); then, we use the l_2 loss

$$\mathcal{L}_{\text{Ego-Motion}} = ||\hat{v} - v^*||_2 \approx \sqrt{(\hat{v} - v^*)^T(\hat{v} - v^*) + \varepsilon} \quad (2)$$

where $\varepsilon = 1.0$ Doppler bins (with a typical magnitude of $16 \leq ||v^*||_2 \leq 32$) is included for numerical stability.

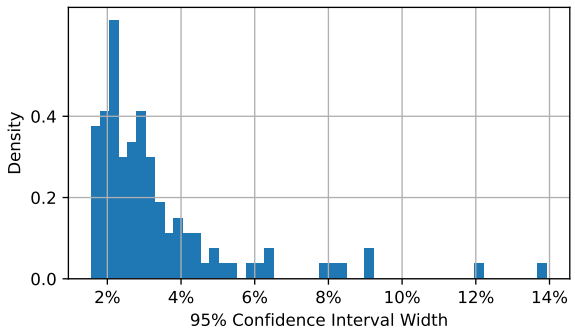


Figure 15. Width of 95% confidence intervals, in percent, relative to the baseline of each ablation, aggregated and plotted as a histogram. Using our 4.5 hour (163k frame) test split, we are able to compare methods with a median confidence interval width of 2.6% (one-sided difference of 1.3%), with the exact width varying depending on the variance of the underlying comparison.

B.4. Evaluation Procedure

Following our evaluation procedure, we can measure differences of 1-2% with high probability (Fig. 15); we provide details about this procedure below.

Geo-Split Within each setting, ≈1.5 hours of data are reserved as a test set. In order to control data leakage, we split traces for each setting along natural geographic boundaries:

- `indoor`: since buildings can have duplicated floor plates and other design features between floors or different areas, data was split by building, with the evaluation set consisting of all traces collected from 3 buildings.
- `outdoor`: each trace was collected as a contiguous area on foot; we reserved a set area within a neighborhood that includes various zoning and street types for the test set.
- `bike`: each trace was collected as a round-trip ride from a set origin; two rides from a set range of directions were reserved for the test set.

Sample Size Correction Intuitively, sampling the same signal – such as radar-lidar-video frames – with a greater frequency yields diminishing "information". Since the standard error of the mean, $\text{SE} = \text{std}(X)/\sqrt{N}$, is defined for N independent and identically distributed samples, we must correct for the *effective sample size* of our test data.

In our analysis, we assuming that changes in model performance imply changes in the underlying data (but not necessarily the converse). This allows us to estimate a lower bound on the effective sample size from each scalar performance metric as [51]

$$N_{\text{eff}} = \frac{N}{1 + 2\sum_{t=1}^{\infty} \rho_t} \tag{3}$$

for autocorrelation ρ_t (where t is the delay). Similar to [18], we approximate the infinite series up to $t = N/2$ and clip negative empirical autocorrelation values $\hat{\rho}_t < 0$ to 0.

Paired z-Test The actual values of each measured metric have a large inherent variance due to the varying difficulty of the data (e.g., the presence of clutter, dynamics, or other challenging features). As such, the standard error of *absolute* values of each metric is large. Taking advantage of the fact that each model is evaluated on identical test traces with respect to a baseline, and that the performance of each model is highly correlated with its baseline, we instead use a paired z-test, i.e., on the *relative* values of each metric, which mitigates the impact of this variance.

B.5. Baseline Details

Ideally, we would like to compare GRT against off-the-shelf baselines without any modifications. However, due to the lack of standardization in radar hardware and modulations, radar data cubes do not have standard dimensions and aspect ratios; furthermore, since radar data cubes are generally tightly coupled with physical effects arising from hardware and modulation design choices, dimension-normalizing transforms such as cropping and resizing are also not generally valid.

While transformer encoder-decoder architectures can be readily adapted for different input and output dimensions by simply changing the input and output context size (and positional encodings), convolutional and encoder-only architectures must be modified to fit different input and output resolutions. Thus, to run prior baselines on our dataset, we made the a few changes to each baseline.

RadarHD RadarHD [45] uses an asymmetric U-net with azimuth-only 2× upsampling layers to meet the target output resolution, relative to the input. Since RadarHD was originally designed for 512 output azimuth bins instead of the 1024 output azimuth bins in our dataset, we include an additional azimuth upsampling layer, with other layers and dimensions staying the same.

T-FFTRadNet T-FFTRadNet [15] uses a swin transformer encoder with a relatively lightweight convolutional decoder, with some U-net-like skip connections. Since T-FFTRadNet's "dense" high-resolution decoder was designed only for cascaded imaging radars, and the decoder for single-chip radar was designed only for sparse outputs, we modified the dense decoder to use the output of the backbone for single-chip radar by increasing the bilinear upsampling size to 4× in both range and azimuth. Other layers and dimensions are kept the same.

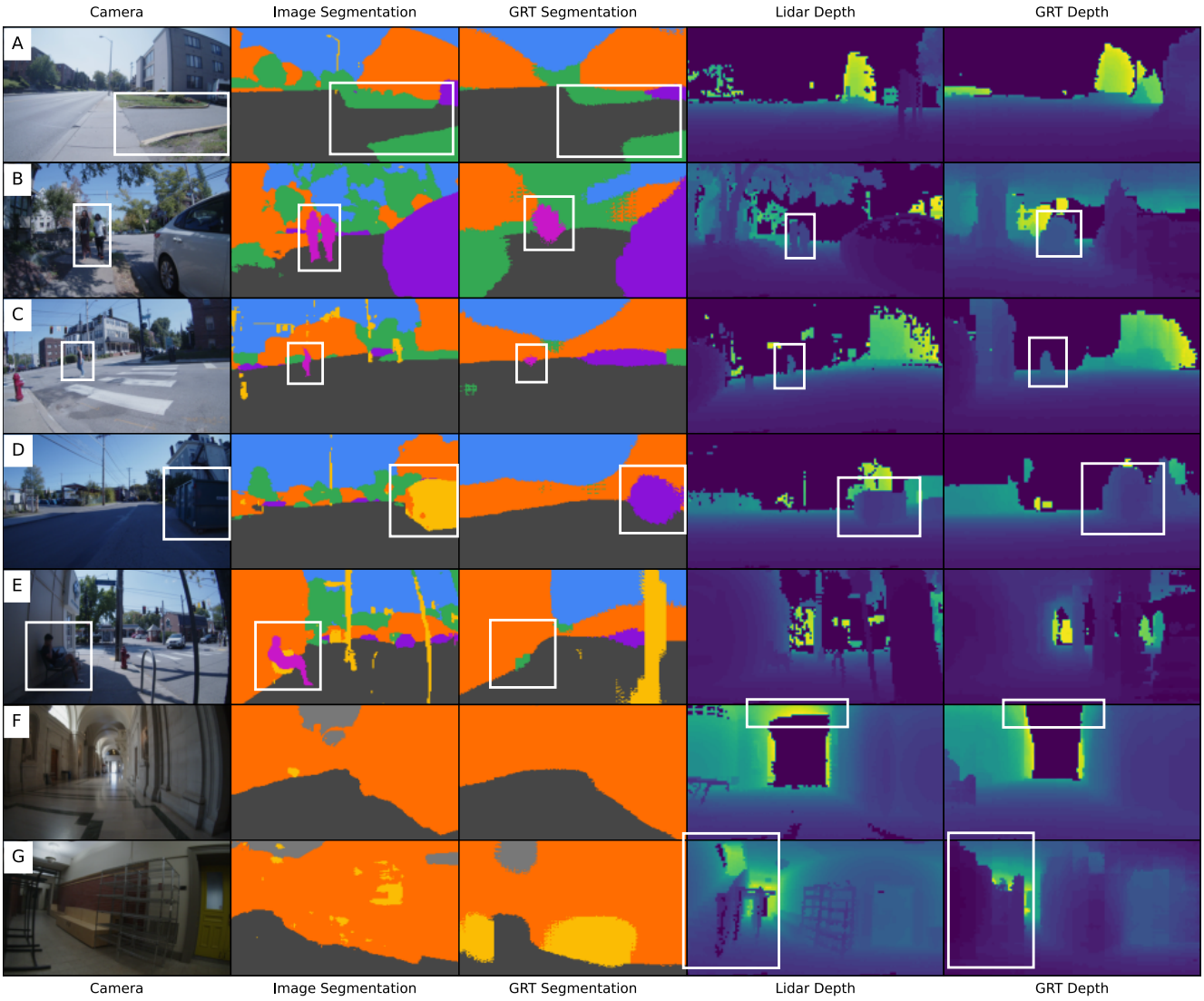


Figure 16. **Select frames from our test set**. Frames are annotated with notable features. Note that the field of view is narrower for the camera ($\approx 60^\circ \times 120^\circ$) compared to the Lidar ($90^\circ \times 180^\circ$).

C. Additional Results

In this section, we provide sample visualizations (App. C.1) and additional analyses (App. C.2-C.4). Note that in addition to the included figures, video examples of our model in action can be found at our project site.

C.1. Sample Images

To better visualize the capabilities of our model, we provide a range of sample results (Fig. 16), including some cases where our model performs better than expected, failure cases which illustrate the limitations of our approach, and a representative random sample (Fig. 17).

Surprising Capabilities While others have tried mmWave-radar-based semantic segmentation [24], no prior works attempt to extract high-resolution elevation information for tasks such as semantic segmentation or 3D occupancy classification on such a low-resolution radar. As such, we found it surprising that our model works at all! In our evaluation traces, we also found additional capabilities which further exceeded our expectations:

- **Material Properties**: Despite our radar's low resolution, it is able to correctly label pavement and grass in many cases (Fig. 16, A), likely learning the fact that paved surfaces generally have specular returns, while natural surfaces have diffuse returns.
- **Pedestrians**: the model is able to correctly identify pedestrians in many cases, likely due to the unique micro-doppler signature of people walking (Fig. 16, B-C).

Finally, it is worth noting that a radar using GRT's segmen-

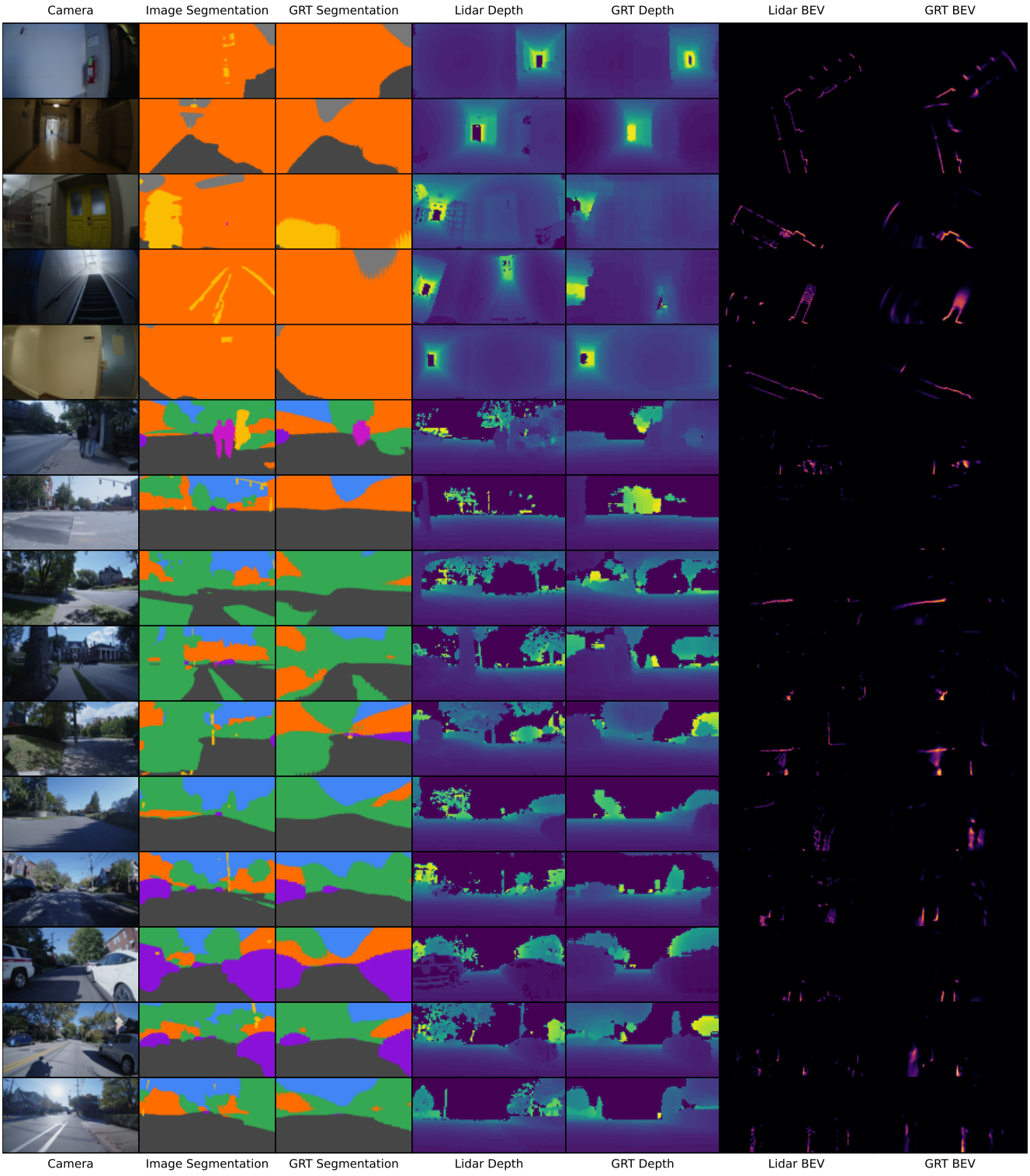


Figure 17. **A random sample of 15 images from the test split of I/Q-1M.** 5 samples are taken from the `indoor` (top), `outdoor` (middle), and `bike` (bottom) settings. All images are generated by `GRT-small`; the segmentation (left) and BEV (right) outputs are trained and fine-tuned on the full dataset, while the depth output (center) is rendered from the base model's 3D occupancy output.

Table 10. Performance metrics (App. B.3) of the `GRT-small` transformer model trained on our base task (3D Occupancy) and fine-tuned on each secondary task with our full dataset (mean with 95% confidence intervals).

Task	Metric	Average	Indoor	Outdoor	Bike
3D Occupancy	Chamfer	4.7 bins $\pm$ 0.19	0.24 m $\pm$ 0.02	0.4 m $\pm$ 0.019	0.38 m $\pm$ 0.023
	Depth	16 bins $\pm$ 0.66	0.62 m $\pm$ 0.059	1.5 m $\pm$ 0.09	1.4 m $\pm$ 0.089
Semantic Segmentation	mIOU	0.69 $\pm$ 0.012	0.78 $\pm$ 0.02	0.63 $\pm$ 0.019	0.69 $\pm$ 0.018
	Accuracy	0.79 $\pm$ 0.0097	0.85 $\pm$ 0.016	0.75 $\pm$ 0.016	0.78 $\pm$ 0.016
	Top-2 Accuracy	0.94 $\pm$ 0.0053	0.97 $\pm$ 0.006	0.92 $\pm$ 0.01	0.93 $\pm$ 0.011
BEV Classification	Chamfer	11 bins $\pm$ 0.91	0.28 m $\pm$ 0.027	0.84 m $\pm$ 0.13	1.3 m $\pm$ 0.19
Ego-Motion Estimation	Speed	0.95 bins $\pm$ 0.093	0.025 m/s $\pm$ 0.0022	0.025 m/s $\pm$ 0.0024	0.091 m/s $\pm$ 0.047
	Angle	4.2° $\pm$ 0.46	5.9° $\pm$ 0.54	5° $\pm$ 0.61	1.9° $\pm$ 1.3

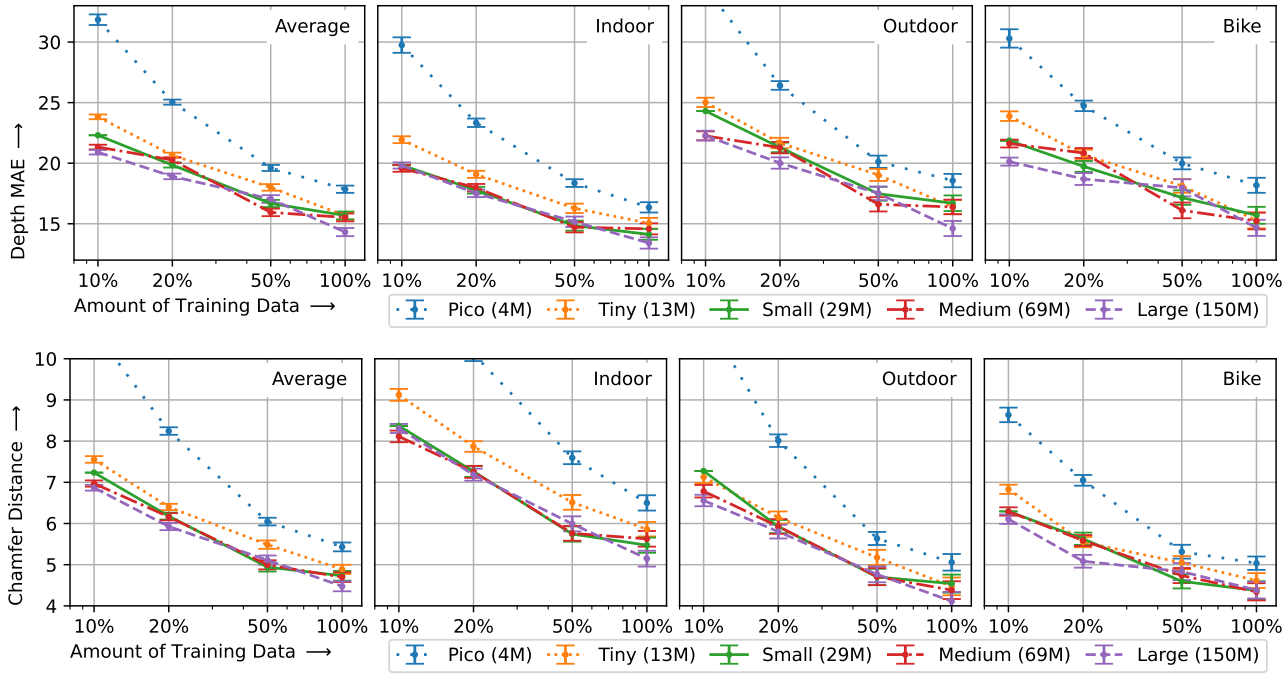


Figure 18. Scaling laws for the *depth* mean absolute error (top) and *chamfer* distance (bottom) metrics, measured in radar range bins (4.4cm `indoor`, 8.7cm `outdoor`, `bike`).

tation capability can operate in conditions when cameras cannot such as fog, smoke, and darkness.

Failure Cases To highlight a few failure cases for GRT:

- **Fine-Grained Classification**: using only a low-resolution radar without any visual or Lidar inputs, a pure radar transformer has no way of differentiating fine-grained classes such as metal dumpsters (in the `object` class) from the `vehicle` class (Fig. 16, D).
- **Static People**: without the unique micro-doppler signature associated with walking, our model often fails to detect people who are standing or sitting still (Fig. 16, E).
- **Limited Vertical Resolution**: the vertical field of view of our radar is relatively limited, with a 6dB-beamwidth of $\pm 20°$. Thus, even with Doppler information to help resolve elevation information (beyond the 2 elevation bins measured by our radar), the model cannot reliably estimate regions at the edge of the Lidar or Camera's vertical field of view (Fig. 16, F).
- **Clutter**: when a scene is very cluttered, GRT can fail to resolve individual objects; this generally leads to large hallucinations (Fig. 16, B, G).

C.2. Absolute Metrics

Since each task has a number of possible metrics (which are not always aligned), we generally report metrics as relative test losses to best capture the performance *of the model* in its ability to fit the target loss.

Key Metrics As an absolute reference, we calculated a range of common performance metrics for each objective

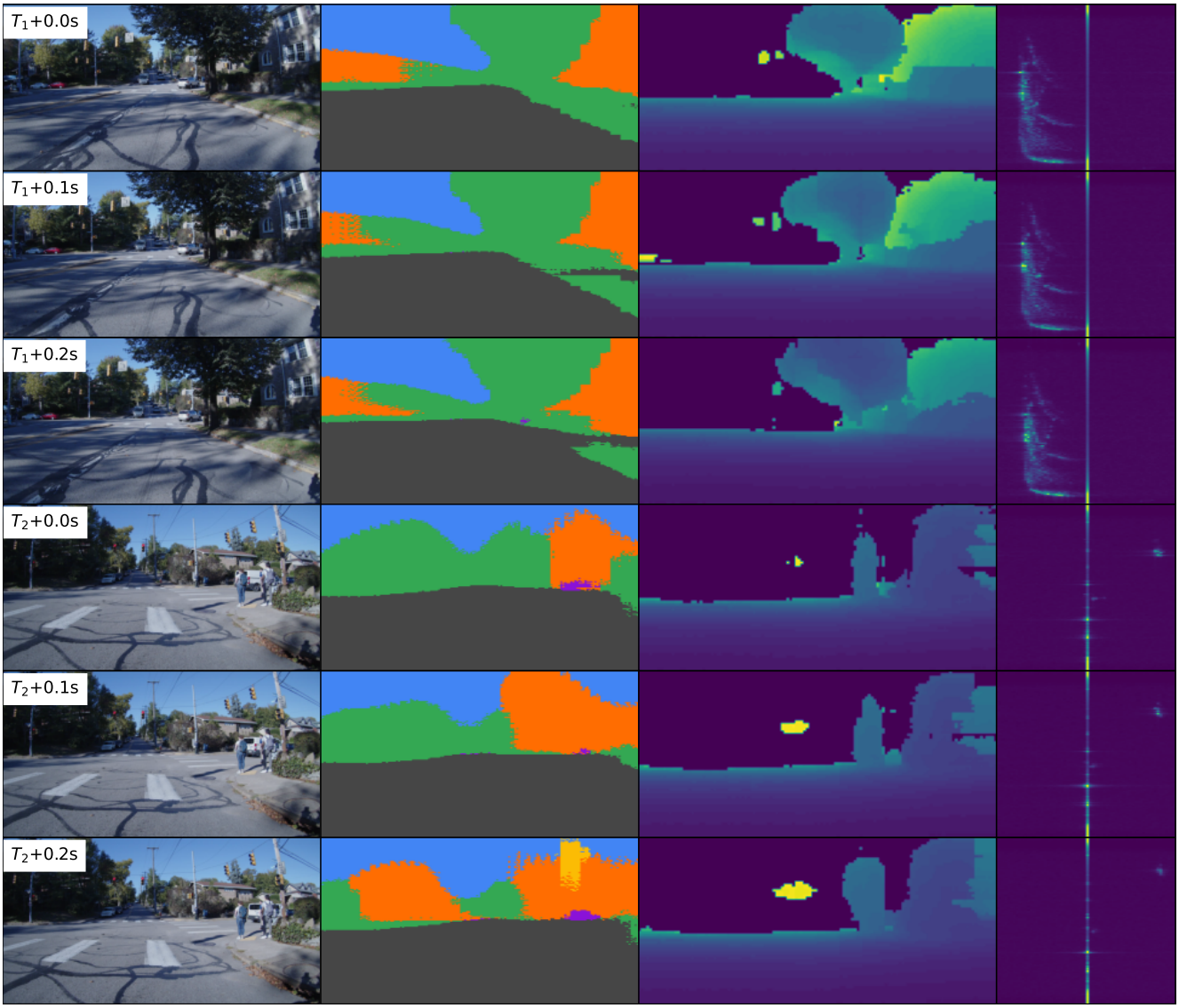


Figure 19. **Sample sequences of three consecutive frames before (top) and after (bottom) stopping at a red light.** Video frames are provided for reference (left), with GRT's semantic segmentation (center left) and depth (center right) outputs across the two sequences along with the range-Doppler spectrum. After stopping, the Doppler spectrum (horizontal axis; right) collapses to a single Doppler bin, resulting in significantly decreased information available to the model. This manifests as noisier (as seen by larger frame-to-frame variations and hallucinations) and less accurate predictions by the model.

(Table 10) as described previously (App. B.3). For these metrics, SI units are reported where applicable; distance and speed metrics are normalized by range and Doppler resolutions, respectively, when aggregating over settings with different radar configurations.

Absolute Scaling Laws As an alternate version to Fig. 6, we also measured scaling laws with respect to the *depth* and *chamfer* metrics (Fig. 18); while somewhat noisier, the same general trend can be seen. Note that this noise is also why we compare loss metrics in our scaling laws and ablations since it serves as a more direct measure of relative "learning" performance.

C.3. Impact of Doppler

To further illustrate the impact of Doppler, we measured the test loss of our objectives (other than Ego-Motion estimation), binned against the sensor speed (Fig. 20). When the sensor's speed is low relative to its maximum Doppler, it captures less Doppler information due to our radar's fixed Doppler resolution, leading to degraded performance. This can also be seen qualitatively: when the data capture rig

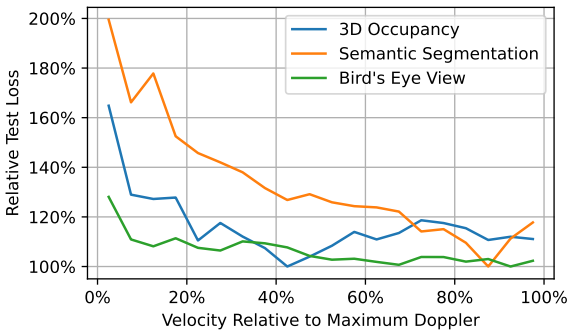


Figure 20. Relative test losses binned by the speed of the data collection rig (20 equal 5% bins) on the `bike` subset; due to less available Doppler information, lower speeds are associated with higher losses for each of our tasks.

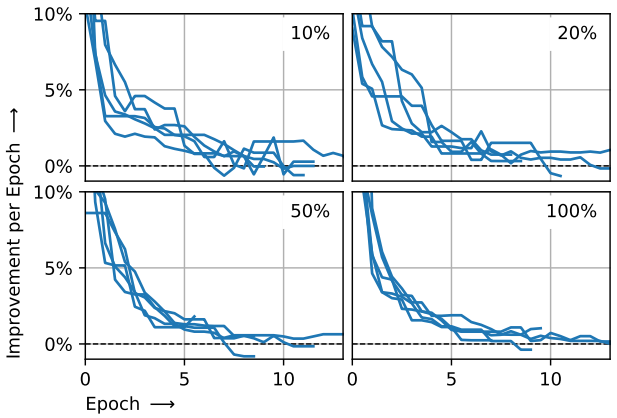


Figure 21. Validation loss improvement per epoch, measured every checkpoint (2 checkpoints/epoch), and smoothed with a 5-checkpoint median filter. Each line refers to a different model (with different sizes); models are separated by dataset size. Our models consistently tend to stop improving (in validation loss) after around 10 epochs of training.

stops, the GRT model's predictions become noisier, less sharp, and tend to show blocky artifacts aligned with the output patch size (Fig. 19).

C.4. Scaling Law Projections

To motivate future work scaling data collection and training for single-chip radar models using 4D data cubes, we run a suite of scaling law experiments to obtain rough, order-of-magnitude estimates for the data requirements of "fully" training GRT foundational model (Sec. 5.4) of a similar size to the relatively small (by vision standards) models which we trained. In this section, we describe additional details and assumptions behind our two methods of estimation.

Extending the Scaling Law In order to estimate data requirements from our scaling law, we start from the assumption that data scaling will always be at most logarithmic. This is based on the intuition that increasing the size of the dataset will always have diminishing returns (in turn at a diminishing rate), which is consistently observed in other scaling experiments [68].

We then train a network only on the test set, without any augmentations; we reason that this provides a lower bound on the achievable test loss (due to random, unpredictable noise in the dataset) for a given architecture, given that the model is not large enough to memorize the test set pixel-for-pixel. Note that this also provides an improvement over a naive lower bound from the fact that $\mathcal{L} > 0$.

Combining these two implies that data scaling can be logarithmic for increasing dataset size up to at most $100\times$ our current dataset size, which we believe represents a reasonable order-of-magnitude estimate for the data requirements for a "fully trained" radar foundational model.

Training Curve Patterns In our experiments, we observe that all models tend to stop improving with respect to validation loss after approximately 10 epochs (Fig. 21). This gives a further avenue for projections: assuming that the informational "value" of a radar frame is roughly equivalent to an image, we can take rough numbers for the typical number of samples seen used to train a vision transformer and translate this to data requirements for a radar foundational model.

Note that this assumption of informational equivalence is also quite rough. Unlike vision transformers, which are typically trained on *independent* images scraped from the internet, GRT is trained using *dependent* frames sampled from a time-series of sensor data, decreasing the relative information density of radar time-series data. On the other hand, while vision transformers typically use sparse feedback signals such as image-caption [47] or image-label [68] pairs, GRT is trained using dense feedback in the form of a 3D occupancy grid (App. B.3). In principle, this increases the relative information density of radar-Lidar training pairs. Our projection therefore assumes that these factors roughly balance out within an order of magnitude.