A Realistic Radar Simulator for End-to-End Autonomous Driving in CARLA

Satyam Srivastava*[*], Jerry Li†[*], Pushkal Mishra‡[*], Kshitiz Bansal‡, Dinesh Bharadia‡

*BITS Pilani, †University of California Riverside, ‡University of California San Diego
srivastavasatyam53@gmail.com, jli793@ucr.edu,
{pumishra, ksbansal, dinesb}@ucsd.edu

***Abstract*—The advancement of self-driving technology is driven by the need for robust perception and navigation systems. Simulators for autonomous driving facilitate the rapid development and testing of navigation algorithms; however, a key issue for most is their inaccurate modeling of the radar sensor. This is a significant drawback as radars offer robust sensing capabilities in adverse weather conditions and occlusions. CARLA, a widely adopted open-source simulator, provides a simplistic radar model that fails to capture the complex physical and material-dependent behavior of real-world radar. To address these limitations, we present C-Shenron, a radar simulation framework integrated into CARLA, which generates realistic radar measurements by fusing LiDAR and camera data. C-Shenron also supports configurable radar parameters, multiple sensor placements, and scalable dataset generation. Our evaluations demonstrate that radar-camera fusion models, trained with C-Shenron's generated data, achieve performance equivalent to traditional LiDAR-camera baselines on key metrics from the CARLA leaderboard.**

***Index Terms*—Radar Simulation, Carla Simulator, End-to-End Driving Models, Multimodal Perception**

I. Introduction

Autonomous systems, especially self-driving cars, rely on End-to-End (E2E) systems that seamlessly connect perception to downstream tasks, such as path planning and navigation. These systems take raw sensor inputs and directly output control actions, making the quality of perception a crucial determinant of overall performance. Robust perception is especially important in complex driving environments, where the ability to accurately detect and track objects directly impacts safety and reliability [1], [2]. While LiDAR and camera sensors have been widely used for perception, their performance can be significantly degraded by adverse weather conditions and occlusions due to the wavelengths they operate in. In contrast, radar sensors, operating with millimeter-wave (mmWave) signals, are highly resilient to adverse weather and lighting conditions, as highlighted in [3].

Additionally, developing and testing E2E autonomous driving systems in the real world is both costly and time-intensive, as even small changes in perception models require collecting and labeling miles of new driving data. This makes iterative development difficult to scale. Moreover, it is challenging to test E2E driving models in real-world settings. Simulators address this bottleneck by offering a controllable and repeatable environment for training and evaluation. The CARLA simulator [4] is a widely adopted open-source platform that facilitates large-scale data collection across diverse handcrafted scenarios, including varying weather and complex traffic conditions. Researchers have extensively utilized CARLA to train and evaluate E2E models as noted in [5]–[7].

[*]Equal contribution.

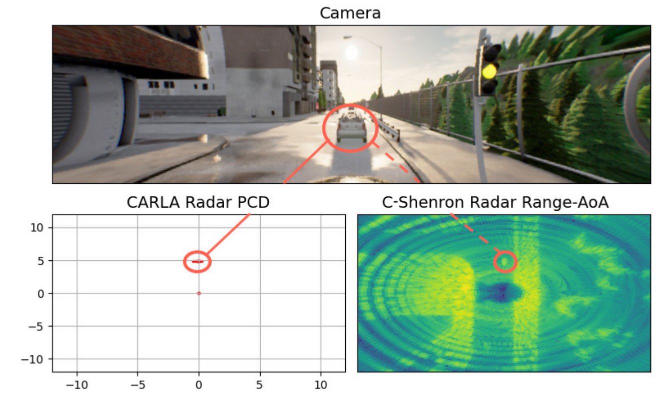


Figure 1. Top image is the camera view from inside the car, bottom plots are the radar point clouds from CARLA and shenron radar, respectively, and both in bird-eye perspective.

However, a significant limitation of CARLA pertains to the modeling of radar sensor returns. Unlike camera or LiDAR, accurately simulating radar is particularly challenging due to the complex physical interactions of millimeter-waves with various materials. CARLA's current radar model simplifies this intricate process, often relying on basic ray-casting methods that fail to account for essential real-world phenomena such as material properties, surface geometry, and signal-level interactions like scattering and reflection. This renders any research involving CARLA radar inadequate, as it does not reflect the real-world capabilities of an operational radar sensor.

In this paper, we present C-Shenron, an innovative radar sensor model integrated into the CARLA simulator. We build on top of the Shenron [8] framework, which fuses LiDAR and Camera measurements via millimeter-wave surface scattering models (refer to Section III-B for more details). The Shenron framework, originally developed and validated with real-world data, is thus brought to CARLA, significantly enhancing the realism of the simulated radar output. A comparison between CARLA's native radar and our enhanced radar is shown in Figure 1. C-Shenron supports a wide range of configurable radar parameters such as number of antenna arrays, chirp time, chirp bandwidth, and experiment with various sensor placements, to explore multiple fusion strategies. This enables a comprehensive multi-modal data collection framework to generate realistic datasets for training and testing of perception models. The contributions of our paper are summarized as follows:

1) **Radar Simulator Integration:** We integrate a realistic radar simulation framework in CARLA, which extends its capabilities by generating physically accurate and material-aware radar data. The nature of the simulator allows the sensor parameters to be configurable, enabling diversity in sensor data generation.
2) **Pipelines for data collection:** We develop efficient pipelines for large-scale data collection and training of deep learning models using the simulated radar data. We have released our codebase, dataset collected and some driving videos here[1].
3) **Benchmarking with E2E Driving models:** We validate the utility of our simulated radar data by integrating it into a state-of-the-art E2E driving model [5]. Our results show that models trained with C-Shenron achieve comparable performance across key CARLA leaderboard metrics, including Driving Score, Route Completion, and Infraction Penalty.

Figure 2 shows a snapshot of the simulated radar, LiDAR and camera data. Camera view is from the driver's perspective, and both LiDAR and radar are in bird-eye view.

II. Related Work

A. Radar Simulators

There have been multiple radar sensor simulators. For single-channel radars, [9] utilized radar range equations combined with a visibility-based approach to model received radar power. But this method does not capture realistic radar scattering effects or extended target behaviors. This paper [10] proposed a method employing the Torrance-Sparrow reflection model, primarily addressing specular reflections from rough surfaces; however, this method similarly neglected comprehensive radar scattering and lacked validation against real-world data.

For MIMO radar simulations, [11] developed a realistic ray-tracing approach, yet its reliance on manually created meshes limits the method's scalability and the ease of integration with existing LiDAR-based datasets. Furthermore, this paper [12] explored GAN-based LiDAR-to-radar translation approaches, but these mainly focus on visual translation rather than physically accurate radar signal simulation. These works highlight a significant gap in the accuracy of radar simulations. In contrast, Shenron [8] performs a much better job at holistically simulating radar data.

B. Real World Radar Datasets

Several large-scale datasets offer diverse radar setups and scenarios to advance radar-based autonomous driving research. nuScenes [13], a widely used benchmark, provides synchronized Camera, LiDAR, and Radar data across diverse urban environments. The Oxford Radar RobotCar Dataset [14] also

[1]Link to website: https://wcsng.ucsd.edu/c-shenron/

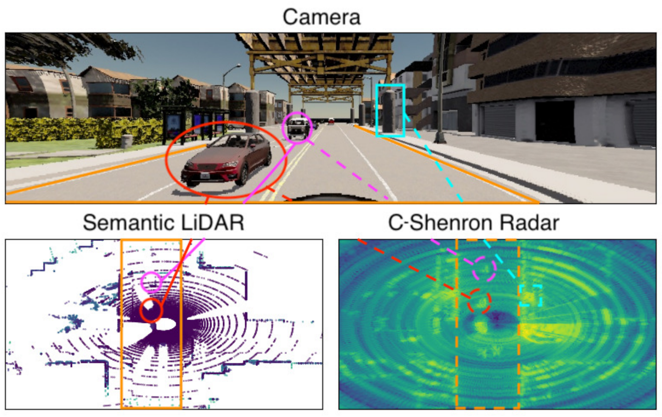


Figure 2. Comparison of views from camera, semantic LiDAR, and shenron radar in CARLA. The orange lines outline the road, red and magenta highlights vehicles, and blue indicates a static object.

offers real-world radar data, specifically targeting challenging weather and lighting conditions. Similarly, the CRUW dataset from RODNet [15] presents synchronized Radar and Camera data collected under varied driving scenarios. Additionally, K-Radar [16] introduces a high-resolution 4D radar benchmark, highlighting radar's robustness compared to LiDAR in adverse weather conditions. But one key problem with these datasets is the inability to perform E2E simulations for testing AD models, hence the need for simulators with realistic radar sensor data.

C. Radar Based Perception

Recent works have focused on enhancing radar-based perception through innovative learning strategies, such as [1] introduced bootstrapping techniques aimed at improving radar performance without the need for extensive manual annotations. This paper [17] further addresses radar's inherent resolution and noise challenges through an adaptive-directional transformer for real-time, multi-view semantic segmentation. Additionally, SIRA [2] contributes scalable inter-frame relation and association methodologies, effectively capturing temporal dynamics to enhance radar-based object detection and tracking. Collectively, these methodologies demonstrate radar's potential as a robust sensor for autonomous driving.

Although multimodal perception leveraging Radar, LiDAR, and Camera sensors has been extensively explored, prominent benchmarks such as the KITTI Vision Suite [18] primarily emphasize Camera and LiDAR-based evaluations, with limited focus on radar-specific assessments. In contrast, our research introduces a realistic radar sensor model integrated into the CARLA simulator, establishes a bridge between radar simulation and real-world applications to facilitate direct evaluation of radar-based end-to-end autonomous driving.

III. Design

The C-Shenron framework, a new, scalable, and highly accurate radar simulator integrated into CARLA. In this section, we describe the radar simulator provided by CARLA, Shenron architecture, and the key computational innovations that make our approach unique.

A. Shortcomings of CARLA Radar model

CARLA is a simulation tool based on Unreal Engine [19], which focuses primarily on generating photorealistic images from cameras. As camera-based simulation is the core focus of CARLA, the radar sensor model it provides is relatively simplistic and lacks physical realism. The radar point cloud is generated simply by the ray-casting methodology, where a user specifies the horizontal and vertical field of view and the number of expected points N. A set of N random ray directions is sampled within the specified field of view, and a ray is cast in each direction. All the points where the rays hit the environment are returned as the final point cloud. This process does not take into account any noise, signal processing, or material properties of the surfaces, hence creating an unrealistic radar point cloud. You can find references here[2].

B. Shenron Primer

Shenron [8] framework simulates a high-fidelity MIMO radar using LiDAR point clouds and camera images. Here, the LiDAR point cloud is used as an impulse response of the environment, representing a real-world ray tracing operation, creating a high-quality representation of the scene without the need for modeling complex geometries. To capture accurate Radio Frequency (RF) reflection profiles for various materials, the framework uses semantic information from the camera images. This allows for material-aware reflectivity modeling, where different surfaces interact with radar waves in a physically consistent manner. This architecture was built and tested in real-world scenarios and showed excellent correlation with actual radar data. As Shenron uses LiDAR and camera as the basis of radar simulation, it makes it a perfect candidate for replacing the existing radar sensor model in CARLA.

C. C-Shenron: CARLA Shenron Integration

CARLA operates on a client-server architecture, where the server simulates the virtual world and the client application interacts with this simulated environment. The server handles the physics simulation, and the collected raw sensor data (camera and LiDAR), along with metadata such as sensor type, frame number, and timestamp, is serialized and transmitted to the client application. The client receives sensor data, processes it, and sends control commands back to the server.

Figure 3 shows the overall picture of the Shenron integration. We devise a hybrid approach by implementing a custom sensor on the server side, which captures the LiDAR Point Cloud (PCD), semantic tags, and relative velocity information for each point in the PCD, along with other metadata, which is further transmitted to the client side. This approach aligns with CARLA's native C++ architecture, ensuring efficient communication and integration with the core simulation loop. The Shenron simulator is run on the client side to generate the simulated radar data. To mitigate the real-time latency introduced by the radar data processing, we paused the CARLA simulation to ensure that the overall simulation time remains unaffected.

[2]GitHub issue: https://github.com/carla-simulator/carla/issues/4974, Documentation: https://carla.readthedocs.io/en/latest/ref_sensors/#radar-sensor

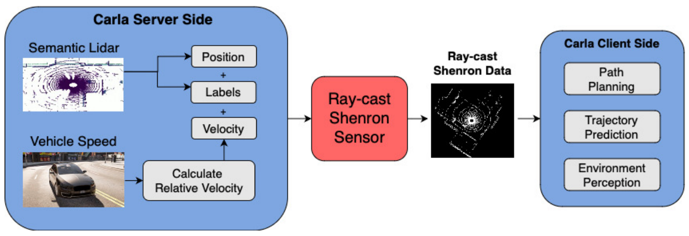


Figure 3. Overall architecture of C-Shenron. We highlight the client-server architecture of the simulator and the necessary data that gets recorded and passed on to the client for the Shenron simulator.

A brief early demonstration of this work was presented as a demo abstract at ACM SenSys'25 [20]. In this paper, we provide a full technical exposition including architectural details, simulation methodology, and extensive evaluation results.

IV. Evaluation Setup

To validate the impact of the C-Shenron Radar simulator for AD tasks, we collect a large-scale dataset with various driving scenarios, integrate a state-of-the-art E2E driving model based on the Transfuser++ [5] architecture, train and evaluate this E2E driving model on multiple driving routes and scenarios.

A. Data Collection

We conducted large-scale data collection using the CARLA Garage platform [21], which uses a rule-based driving expert [22], just like an experienced human driver, thus producing high-quality driving data. This expert follows predefined traffic rules, accesses map data (e.g., lane boundaries, signals, speed limits), and plans routes with high precision.

Our dataset includes a variety of environments, weather conditions, and sensor configurations to create a comprehensive evaluation benchmark. We collect data across 8 CARLA towns (Town01 - Town07 and Town10) under multiple driving scenarios such as urban cities with dense streets, residential areas with suburban roads, and open highway settings. Also, we include multiple weather types, such as rainy, foggy, and nighttime conditions. To accelerate data collection, multiple CARLA instances are launched in parallel, using a Kubernetes cluster to launch multiple jobs, which reduces the collection time from days to hours. This results in 70 unique combinations, with each combination repeated three times, yielding a total of 850,000 frames. The code and dataset are present in the GitHub repository mentioned in the introduction section.

B. Integrating with Transfuser++ Architecture

The Transfuser++ [5] architecture is an imitation driving model that uses a robust transformer-based sensor fusion module integrating camera and LiDAR data, alongside auxiliary branches for perception tasks like classification, detection, and segmentation. The model is trained to predict the steering angle and acceleration required for the car to drive safely based on the Camera and LiDAR inputs. For more details on the architecture, refer to the paper [5].

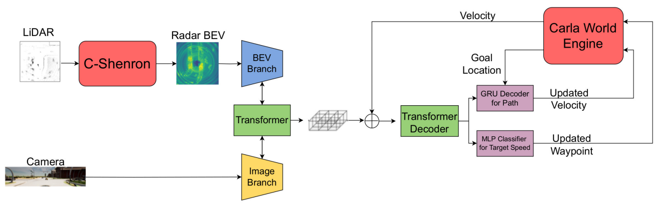


Figure 4. C-Shenron with the Transfuser++ Architecture

In our implementation, radar data is generated as range-angle plots and fed directly as images into the BEV branch, bypassing the LiDAR images. This is depicted in Figure 4.

C. Training details

The data generated from the expert driver (Section IV-A) contains LiDAR, radar, camera, acceleration, and steering data, which is directly used to train the model. Training includes the same loss function as in [5], along with other parameters such as batch size of 12, 30 epochs, and a learning rate of 3x10^{-4}. We trained the model on a system equipped with 6 NVIDIA A10 GPUs, which required approximately 2 days to complete on the entire 855,000 frames of data.

D. Evaluation Metrics

The driving proficiency of an autonomous agent is evaluated through various metrics in CARLA[3], providing insights into different aspects of its driving behavior. For our setup, the metrics used are described as follows:

- **Route completion**: It is the percentage of the route distance completed by an agent. The higher the percentage, the better it is at driving through a given scenario. For a given Route i, it is denoted as C_i.
- **Infraction Penalty**: CARLA tracks multiple types of infractions, and each infraction has a penalty score between 0 and 1 depending on the severity of the infraction. Furthermore, these scores are consolidated into a single score through a geometric series, based on the number of infractions and the type of infractions committed by the agent. If for the Route i, n_j and p_j are the number of infractions and infraction score for the j^{th} infraction committed by the agent, then the infraction penalty is calculated as:
$$P_i = \prod_j p_j^{n_j}$$
- **Driving Score**: This is the overall score for the agent, calculated per route and averaged across all routes. For n routes, the driving score is calculated as:
$$\text{DS} = \frac{1}{n}\sum_{i=1}^{n} C_i P_i$$

Infractions within the CARLA simulation environment are penalized based on their severity, with specific coefficients assigned to various types of infractions. For instance, collisions with: pedestrians (0.50), other vehicles (0.60), static objects (0.65), traffic violations such as running a red light (0.70), running a stop sign (0.80), and many more.

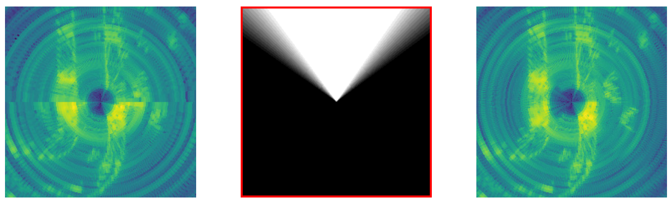
V. Results

The driving models are evaluated on the routes from NEAT [6], which include various settings like highways, urban areas, and residential zones with diverse road layouts and obstacles to simulate realistic conditions. Agents face traffic scenarios based on NHTSA typology[4], such as navigating intersections, responding to pedestrians, cyclists, and other road users. To ensure consistency, each model was tested on the same set of 14 routes over 5 iterations under stable conditions without extreme weather. Additionally, we carried out three case studies to examine the impact of each radar view, different sensor placements, and radar resolution on E2E driving tasks. Note that in all the evaluations, camera data was always fed into the model.

A. Does increasing radar views help?

Here, we analyze the effects of increasing the number of radar views on our autonomous vehicle. The Shenron radar generated from combining Camera and LiDAR offers a 180° field of view (FOV), but the image quality decreases as the coverage angle widens. We evaluate three configurations of our radar models: front-only radar, front + back radars (denoted as FB), and full coverage with front + back + left + right radars (denoted as FBLR). All configurations are also fused with camera features. Note that all the views of radar have 180° FOV.

Combining the radar views in the FB scenario is straightforward; the two can simply be concatenated vertically to create a complete 360° image, as illustrated in Figure 5a. However, an interesting challenge arises when attempting to merge the four radar views into a single high-quality image.



(a) FB concat (b) Mask for FBLR (c) FBLR concat

Figure 5. Images representing: (a) Radar image after FB concatenation, (b) Mask for FBLR concatenation, (c) Radar image after FBLR concatenation.

We devise a masking approach wherein the overlapping border regions from different views are used to average the inconsistency and create a cohesive view of all four radars. The specialized mask, as shown in Figure 5b, is rotated for proper orientation, applied to each radar view, and further combined through pixel-wise addition. The mask's magnitude decreases linearly before the ±45° line and drops to 0 beyond the line, which compensates for brightness variation in the overlapping regions when performing pixel-wise addition. The resulting

[3]More information can be found here: https://leaderboard.carla.org/

[4]https://www.nhtsa.gov/sites/nhtsa.gov/files/811731.pdf

composite radar image, Figure 5c, demonstrates the efficacy of this approach and creates an accurate representation of the vehicle's surroundings.

The findings are presented in Table I. LiDAR serves as the baseline for comparison, being the original version of Transfuser++ retrained on the collected LiDAR and Camera data using the same parameters. The expert model represents statistics from CARLA's driver agent, which sets a theoretical upper-performance limit, as the training data was derived from this agent.

Radar View	RC ↑	IS ↑	DS ↑
LiDAR [5]	95.93 ± 3.43	0.79 ± 0.05	76.84 ± 5.26
Front	95.12 ± 3.02	0.82 ± 0.06	78.14 ± 3.7
FB	**96.51 ± 2.99**	0.79 ± 0.02	78.26 ± 2.96
FBLR	93.56 ± 2.75	**0.84 ± 0.05**	**79.24 ± 1.85**
Expert	97.394	0.964	93.82

Table I
RESULTS FOR DIFFERENT RADAR VIEWS WITH ROUTE COMPLETION (RC), INFRACTION SCORE (IS), AND DRIVING SCORE (DS).

Among the radar models, the FBLR configuration demonstrates the best performance in driving score and infraction score. It also has the lowest variance in driving score, indicating that additional field-of-views enhance consistency and improves situational awareness of the model. Although none of the models achieve expert performance, the FBLR radar configuration is the closest across all metrics. Overall, radar-based models showcase equivalent performance compared to LiDAR-camera setups in all key areas, demonstrating the viability of the simulated data.

B. Redaction of Radar views

To evaluate the utility of each radar sensor placement in the FBLR model, we conduct an ablation study where one of the four radar views is removed at a time and re-run the simulation for each configuration. This approach helps to assess the impact of each radar placement on the overall driving performance.

Redact	RC ↑	IS ↑	DS ↑
Left	93.65 ± 2.68	0.78 ± 0.02	75.79 ± 1.79
Right	91.06 ± 0.91	**0.82 ± 0.04**	**76.61 ± 3.00**
Front	91.07 ± 3.66	0.37 ± 0.10	35.88 ± 8.63
Back	**96.16 ± 3.80**	0.77 ± 0.03	73.30 ± 4.25
No Redact	93.56 ± 2.75	0.84 ± 0.03	79.24 ± 1.85

Table II
REDACTION OF RADAR RESULTS WITH ROUTE COMPLETION (RC), INFRACTION SCORE (IS), AND DRIVING SCORE (DS). NO REDACT IS THE SAME RESULT FROM TABLE I AND IS MENTIONED HERE FOR COMPARISON ONLY.

The results from Table II indicate that redacting the front view results in the most significant drop in performance, suggesting that the front view is critical for obstacle detection and lane positioning. In contrast, redacting left or right views has a smaller impact on performance, indicating that while these views contribute to lateral awareness, they are less crucial than the front view. Similar results are also observed for removing the back radar view. The camera-only model performs the least

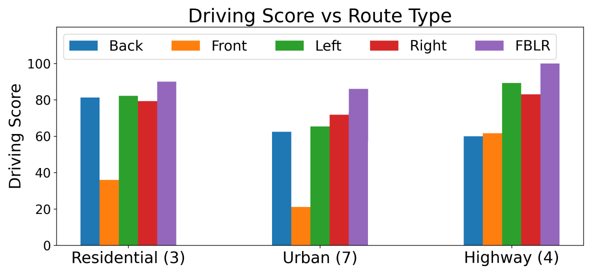


Figure 6. Route-wise driving score for radar redactions. The three categories have 3, 7, and 4 routes, respectively. The FBLR scores are the original scores, i.e., without redacting any of the radar views, and are plotted for reference only.

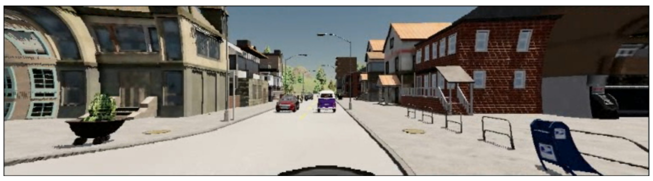
across all the scores, indicating that having radar views helps the model.

Route-wise scores from Figure 6 solidify the point that combining all four views gives optimal situational awareness in the FBLR model. Throughout all routes, the redaction of the front view consistently scores lower, suggesting it is more critical than other perspectives.

C. Experimenting with Radar Resolution

The number of antennas determines the resolution of a radar sensor; more antennas yield higher resolution but also increase cost. C-Shenron supports fully configurable radar parameters, allowing simulation of realistic radar data across a wide range of sensor configurations. In this case study, we reduce the number of antennas in the receiver array from 86 (TI Cascade radar) to 16 (INRAS Radarbook) and demonstrate C-Shenron's ability to generate highly accurate radar data even under such constraints, highlighting its flexibility and realism.

The resulting radar views are seen in Figure 7, where both the low and high-resolution radar views are generated for the same scene of the vehicle, making it clear that the latter configuration has a higher angular resolution than the former radar configuration. The resulting radar data (16 antennas)


(a) Camera View

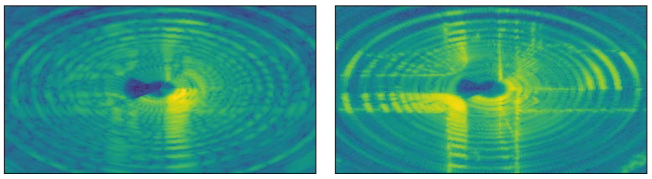

(b) Low Resolution Radar View (c) High Resolution Radar View

Figure 7. Comparison of radar image for a given scenario: (a) Camera View, (b) Radar view with 16 linear antenna array, (c) Radar view with 86 antenna array.

was used for training and evaluation of Front, FB, and FBLR models, and the results are provided in Table III.

Radar View	**RC** ↑	**IS** ↑	**DS** ↑
Front	91.56 ± 2.26	**0.79 ± 0.04**	**73.82 ± 4.94**
FB	**92.61 ± 0.94**	0.75 ± 0.07	72.75 ± 6.85
FBLR	80.69 ± 4.65	0.64 ± 0.06	54.23 ± 5.84
FBLR (86x)	93.56 ± 2.75	0.84 ± 0.05	79.24 ± 1.85

Table III
RESULTS FOR DIFFERENT RADAR VIEWS USING 16-ANTENNAS WITH ROUTE COMPLETION (RC), INFRACTION SCORE (IS), AND DRIVING SCORE (DS). THE 86 ANTENNA RESULTS ARE USED AS A REFERENCE HERE.

As seen from the table, the high-resolution FBLR (86x) model achieves much better results when compared to low-resolution radar configurations, mainly because of having fewer infractions (higher infraction score). Also, increasing the number of radar views paradoxically degrades performance, as evidenced by the FBLR having substantially lower driving scores. This can be attributed to the blurry and imprecise nature of low-resolution radar views, which becomes problematic when multiple views are stitched together, as is also evident by the front-only model having a higher score than other models.

VI. LIMITATIONS & FUTURE WORK

In this work, we introduced C-Shenron, a high-fidelity radar simulation framework integrated into the CARLA simulator, enabling realistic, physics-based radar data generation for E2E AD research. The key limitation to our approach is that the simulator requires additional GPU compute and frequent pauses in the simulation time, effectively requiring higher GPU computations and larger simulation times.

Looking ahead, we plan to expand our experiments by incorporating longer and more diverse driving scenarios across additional CARLA towns and comparing performance with a broader set of state-of-the-art driving models. C-Shenron lays the groundwork for radar-first autonomous driving research in simulation and opens the door to more robust and scalable AD systems.

REFERENCES

[1] Yiduo Hao and et al., "Bootstrapping autonomous driving radars with self-supervised learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[2] Ryoma Yataka, Pu Wang, Petros Boufounos, and Ryuhei Takahashi, " SIRA: Scalable Inter-Frame Relation and Association for Radar Perception ," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Los Alamitos, CA, USA, June 2024, pp. 15024–15034, IEEE Computer Society.

[3] Kshitiz Bansal, Keshav Rungta, Siyuan Zhu, and Dinesh Bharadia, "Pointillism: Accurate 3d bounding box estimation with multi-radars," in *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, 2020, pp. 340–353.

[4] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun, "Carla: An open urban driving simulator," 2017.

[5] Bernhard Jaeger, Kashyap Chitta, and Andreas Geiger, "Hidden biases of end-to-end driving models," in *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2023.

[6] Kashyap Chitta, Aditya Prakash, and Andreas Geiger, "Neat: Neural attention fields for end-to-end autonomous driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15793–15803.

[7] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger, "Transfuser: Imitation with transformer-based sensor fusion for autonomous driving," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 11, pp. 12878–12895, 2022.

[8] Kshitiz Bansal, Gautham Reddy, and Dinesh Bharadia, "Shenron - scalable, high fidelity and efficient radar simulation," *IEEE Robotics and Automation Letters*, vol. 9, no. 2, pp. 1644–1651, 2024.

[9] X. Li et al., "Research on millimeter wave radar simulation model for intelligent vehicle," *Int. J. Automot. Technol.*, vol. 21, no. 2, pp. 275–284, 2020.

[10] T. Machida and T. Owaki, "Rapid and precise millimeter-wave radar simulation for adas virtual assessment," in *Proc. of the IEEE Intell. Transp. Syst. Conf.*, 2019, pp. 431–436.

[11] C. Schüssler, M. Hoffmann, J. Bräunig, I. Ullmann, R. Ebelt, and M. Vossiek, "A realistic radar ray tracing simulator for large mimo-arrays in automotive environments," *IEEE J. Microw.*, vol. 1, no. 4, pp. 962–974, Oct 2021.

[12] L. Wang, B. Goldluecke, and C. Anklam, "L2r gan: Lidar-to-radar translation," in *Proc. Asian Conf. Comput. Vis.*, 2020.

[13] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qing Xu, Anush Krishnan, 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*, 2020, pp. 12165–12175.

[14] 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)*. 2020, pp. –, IEEE.

[15] 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*, 2021, pp. 504–513.

[16] Dong-Hee Paek, SEUNG-HYUN KONG, and Kevin Tirta Wijaya, "K-radar: 4d radar object detection for autonomous driving in various weather conditions," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds. 2022, vol. 35, pp. 3819–3829, Curran Associates, Inc.

[17] Yahia Dalbah, Jean Lahoud, and Hisham Cholakkal, "Transradar: Adaptive-directional transformer for real-time multi-view radar semantic segmentation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024.

[18] Andreas Geiger, Philip Lenz, and Raquel Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3354–3361.

[19] Epic Games, "Unreal engine," https://www.unrealengine.com.

[20] Pushkal Mishra, Satyam Srivastava, Jerry Li, Kshitiz Bansal, and Dinesh Bharadia, *Demo Abstract: C-Shenron: A Realistic Radar Simulation Framework for CARLA*, p. 726–727, Association for Computing Machinery, Irvine, CA, USA, 2025.

[21] Autonomous Vision Group, "Carla garage," https://github.com/autonomousvision/carla_garage, 2024.

[22] Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Jens Beißwenger, Ping Luo, Andreas Geiger, and Hongyang Li, "Drivelm: Driving with graph visual question answering," in *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part LII*, Berlin, Heidelberg, 2024, p. 256–274, Springer-Verlag.