A Realistic Radar Simulation Framework for CARLA

Anonymous CVPR submission

Paper ID 12354

Abstract

The advancement of self-driving technology has become a focal point in outdoor robotics, driven by the need for robust and efficient perception systems. This paper addresses the critical role of sensor integration in autonomous vehicles, particularly emphasizing the underutilization of radar compared to cameras and LiDARs. While extensive research has been conducted on the latter two due to the availability of large-scale datasets, radar technology offers unique advantages such as all-weather sensing and occlusion penetration, which are essential for safe autonomous driving. This study presents a novel integration of a realistic radar sensor model within the CARLA simulator, enabling researchers to develop and test navigation algorithms using radar data. Utilizing this radar sensor and showcasing its capabilities in simulation, we demonstrate improved performance in end-to-end driving scenarios. Our findings aim to rekindle interest in radar-based self-driving research and promote the development of algorithms that leverage radar's strengths.

1. Introduction

Autonomous systems, especially self-driving cars, rely on end-to-end pipelines that seamlessly connect perception to downstream tasks like path planning and navigation. While robust perception is a critical component of these systems, the focus in end-to-end approaches is on ensuring that sensor data directly informs actionable decisions. Multimodal sensor fusion plays a pivotal role in this context, enabling a holistic understanding of the environment by integrating complementary inputs from camera, LiDAR and radar [8, 18]. This fusion enhances the system's resilience to varying conditions- radar excels in detecting speed and distance in adverse weather, while camera offers detailed visual information for interpreting road signs and traffic signals [25].

Making multimodal sensors work well together also requires a detailed understanding of how each sensor operates, including their strengths, limitations, and behavior under different conditions. Expanding on this understanding, a fundamental question lies in determining what the right configuration and placement of sensors are, enabling low cost while ensuring robust performance and appropriate sensor fusion algorithms to enable safe perception, navigation, and path planning. Building all different configurations and hardware to achieve these objectives is impossible, highlighting the need for simulation tools. In addition, training such perception models for autonomous driving requires significant amounts of data encompassing various scenarios to ensure reliable performance under different conditions [2, 12, 17, 20]. A key challenge here is large-scale data collection, as collecting data for every possible situation is nearly impossible. Moreover, the collected data is significantly impacted by the way the sensors are placed and their specific characteristics. This challenge emphasizes the potential of simulations to enhance real-world data collection.

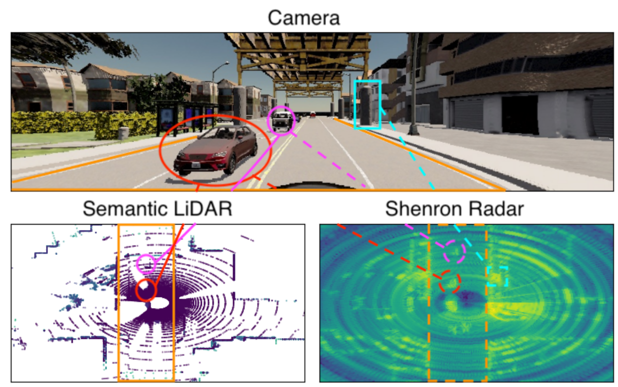


Figure 1. Comparison of views from Camera, Semantic LiDAR, and Shenron Radar in CARLA. The orange lines outline the road, red and magenta highlights vehicles, and blue indicates a static object.

The CARLA simulator excels in enabling both perception and downstream tasks in autonomous driving research. It facilitates large-scale data collection by generating diverse datasets that capture a wide range of scenarios, including varying weather conditions and complex traffic environments [11]. CARLA supports end-to-end training pipelines by providing accurate simulation of key sensors like camera and LiDAR, making it an effective digital twin for the rapid development and testing of autonomous systems [1]. Researchers have extensively utilized CARLA to train perception models and integrate them into downstream tasks like path planning and navigation, as noted in works such as [7, 8, 16]. The simulator's flexibility and precision have solidified its role as a vital tool for testing and validating state-of-the-art approaches, particularly in systems that leverage multi-modal fusion to achieve robust and reliable performance.

While LiDAR is a useful sensor, it struggles with all-weather sensing due to its reliance on lasers. In contrast, radar employs millimeter-wave technology and is highly effective in various conditions[3]. However, the radar model in the CARLA simulator has significant limitations. Unlike real-world radar systems that utilize multiple radar beams, advanced Doppler processing, and sophisticated clutter filtering, CARLA's radar is a simplified version that lacks these essential features. It generates data by randomly sampling LiDAR outputs, failing to capture key radar-specific characteristics, such as sensitivity to motion and environmental influences. Additionally, there have been multiple velocity computation issues, with moving vehicles displaying inaccurate speed readings [10]. These shortcomings render any research involving CARLA radar inadequate, as it does not reflect the real-world capabilities of an operational radar sensor that can be used in autonomous vehicles [21].

In this paper, we present C-Shenron, an innovative radar sensor model integrated into the CARLA simulator, extending the Shenron framework, which previously focused solely on LiDAR data [4]. C-Shenron allows users to configure and simulate diverse radar setups with different number of antenna arrays, thereby enabling comprehensive multi-modal data collection and simulation for end-to-end autonomous driving tasks. With C-Shenron, researchers can experiment with various radar sensor placements, explore multiple fusion strategies, and generate high-fidelity datasets for training and testing robust perception models.

To achieve seamless functionality, we designed a server-side sensor in CARLA that aggregates required data from the simulation world into a unified stream, enabling efficient radar data generation and fusion with Shenron existing capabilities. This innovation bridges the gap between CARLA and Shenron, establishing a cohesive platform for advancing radar-based multimodal fusion research in autonomous driving research.

To demonstrate the functionality of this new sensor, we gathered data, trained, and evaluated the model within the CARLA simulator. We are also the first to generate high quality radar data across various towns and scenarios, utilizing Kubernetes for automation and scaling. The data generated from the integrated radar sensors and camera was then utilized to train a state-of-the-art model [16], improving the perception capabilities of the framework. This comprehensive training showcased the benefits of multimodal fusion to achieve accurate and reliable driving in a realistic simulation.

We evaluated the end-to-end model in diverse driving scenarios in a simulated environment. Using the simulator allowed us to position various radars on the vehicle to identify the optimal setup for driving performance. Another significant challenge was to integrate multiple radar views to achieve one 360° radar image to provide comprehensive situational awareness. We implemented a masking procedure to stitch these views together which enhanced our model's situational awareness. We also evaluate of each radar view's utility through a redaction process, ensuring the model accurately interpreted the combined radar information. Our results highlight that radar and camera-based models achieve better performance in some scenarios and comparable performance in others, compared to traditional camera and LiDAR models.

The remainder of the paper is structured as follows: we review related work, discuss how radar enhances autonomous driving reliability alongside CARLA, detail the design and implementation of our approach, and conclude with evaluations and future work proposals.

2. Related Work

The development of sensor technologies for autonomous driving has predominantly focused on vision-based and LiDAR-based perception systems, attributed to their high-resolution capabilities and the availability of extensive datasets.

Vision-Based Perception: Camera-based approaches have gained widespread adoption for tasks such as object detection, lane detection, and scene understanding. The success of these methods is largely due to the availability of large-scale datasets like KITTI, Cityscapes, and nuScenes, which facilitate the training of robust computer vision models [13]. These datasets have enabled rapid advancements in visual perception algorithms, leveraging deep learning architectures to achieve high accuracy in identifying objects, detecting obstacles, and recognizing traffic signs and signals [9].

LiDAR-Based Perception: LiDAR technology is also prevalent in autonomous vehicle research due to its precise depth information and accurate 3D mapping capabilities. This allows for complex tasks such as 3D object detection and point-cloud segmentation. Significant advancements in LiDAR-based perception have been supported by dedicated datasets like the Waymo Open Dataset and SemanticKITTI [22]. These resources, combined with LiDAR's ability to capture detailed 3D spatial information, have made it a preferred choice for high-resolution sensing in self-driving systems. However, LiDAR performance can degrade in adverse weather conditions and struggles with occlusion penetration, posing challenges in real-world scenarios [5].

Radar-Based Perception: Radar technology has emerged as a crucial component in the sensor suite for autonomous vehicles. Sensor fusion techniques have been pivotal in enhancing radar-based perception by integrating data from multiple sensors, including lidar and cameras. This multi-modal approach leverages the strengths of each sensor type to improve detection accuracy and robustness [8]. Studies have shown that fusing radar data with visual information can significantly enhance performance in complex driving scenarios by providing complementary information that addresses individual sensor limitations [22].

A novel approach proposed by Kshitiz et al. [3] enhances radar-based perception by employing multiple radar units to generate accurate 3D bounding boxes for object detection. Another work by Kshtiz et al. [4] laid the groundwork for developing realistic radar sensing models, which we extend in this paper to enhance the CARLA simulator. However, challenges remain, such as dealing with sparse data and optimizing algorithms to better interpret radar measurements under varying conditions. By integrating a high-fidelity radar model, we aim to open new avenues for self-driving algorithms that utilize radar data effectively.

The CARLA simulator, which stands for CAR Learning Algorithm, has facilitated numerous advances in autonomous driving research by providing robust support for various sensors[6, 7, 15, 19, 24]. However, the lack of realistic radar sensor simulations within CARLA limits its utility for research focused on radar-based navigation [11].

Multi-Modal Sensor Fusion: The introduction of the TransFuser model [8] in 2021 marked a significant step forward in multi-modal sensor fusion approaches for autonomous driving. Utilizing a transformer architecture for end-to-end driving policy development, TransFuser integrates data from cameras and LiDAR to enhance performance in complex driving scenarios. By effectively combining these diverse sensor inputs, it addresses the limitations inherent to single-sensor approaches. TransFuser++ [16] builds upon this foundation with improved sensor integration and advanced data augmentation techniques. It introduces cross-attention mechanisms that better align inputs from different sensors, addressing compounding errors in trajectory prediction. By incorporating updated training protocols and data handling strategies, TransFuser++ achieves higher performance benchmarks, such as CARLA's Longest6 and MAP leaderboard, demonstrating its capability to maintain route accuracy while reducing infractions.

This evolution underscores the potential of multi-sensor fusion approaches in designing more resilient autonomous driving systems that can integrate new sensors like radar to enhance perception and decision-making.

3. Background

3.1. Radar in Autonomous Driving

In the real world, Camera and LiDAR are more commonly used in autonomous driving than radar due to radar's inconsistent standardization and its sensitivity to noise and lower resolution. However, Radar offers unique benefits compared to LiDAR and cameras, especially in adverse weather conditions. Unlike optical sensors, radar uses radio waves, allowing it to penetrate through rain, fog, snow, and dust, making it more reliable for all-weather performance. Its long-range detection capabilities, as noted in Table 1, surpass those of LiDAR and cameras, which is particularly useful in high-speed driving and congested environments. Additionally, radar's ability to maintain low noise sensitivity and track velocity over long distances, as shown in Table 1, highlights its suitability for challenging driving scenarios. Radar's doppler measurement capability, which provides information on the relative velocity of objects, is crucial for tasks like path planning, trajectory prediction, and enhancing spatial resolution.

3.2. CARLA Sensors

Sensors act as the eyes and ears of autonomous vehicles, making it crucial for the CARLA simulator to provide accurate and realistic sensor simulations. CARLA includes all the main sensors needed for autonmous driving such as camera, LiDAR, radar, GNSS (Global Navigation Satellite System), IMU (Inertial Measurement Unit) and many others. Furthermore CARLA includes sensors that are challenging to access in real-world scenarios due to safety and logistical constraints, such as collision and lane invasion detectors, an odometer, and a Road Surface Sensor (RSS) that communicates traffic signals and lane markings.

3.3. Unrealistic Qualities of CARLA Radar

CARLA provides researchers with a unique opportunity to access high-quality multi-sensor data, which is often challenging to obtain in real-world environments. However, the default radar sensor in CARLA has limitations that hinder its performance in tracking objects behind other vehicles and in long-range obstacle detection scenarios. It only provides point cloud data for detection and tracking, lacking real-time velocity information, which is essential for accurately assessing object motion and ensuring safe navigation. While point cloud data allows precise mapping through 3D coordinates, the absence of velocity data forces reliance on

Sensor Type	Cost	Noise Sensitivity	Range	Resolution	Weather Resistance	Velocity Tracking	Height Tracking
Camera	✓	✓	●	✓	×	×	×
LiDAR	×	×	✓	●	×	●	●
Radar	✓	✓	✓	×	✓	✓	✓

Table 1. Comparison of sensor types—Camera, LiDAR, and Radar—across various attributes. Green checkmarks indicate favorable traits, yellow circles indicate moderate traits, and red crosses indicate unfavorable traits.

historical position data, which can result in delayed reactions and reduced situational awareness. Furthermore, raw 3D radar data provides a richer, more detailed representation of the environment compared to traditional radar point cloud data, making it particularly valuable for applications in autonomous driving and advanced perception systems. Our proposed C-Shenron radar provides high-quality, accurate radar data.

4. Design

We integrate a new scalable, high-fidelity, and efficient radar (Shenron) sensor with the CARLA simulator. Shenron is an open-source framework that can simulate high-fidelity MIMO radar data using the information from the LiDAR point clouds and camera images. It leverages the impulse response captured by LiDAR sensors, which provide a point cloud representation of the environment, to simulate radar data without the need for complex geometries. To derive accurate radio frequency (RF) reflection profiles for various materials, the framework uses semantic information from the camera images. By combining both specular and scattering reflection models, Shenron achieves a high correlation with real-world radar data, making it a robust tool for evaluation of radar algorithms.[4].

Shenron requires lidar point cloud data, along with semantic tags and the relative velocity of those points concerning the sensor, as input to generate raw 3D radar data, which includes range, angle, and doppler dimensions. The new sensor we introduce on the server side of CARLA fulfills these requirements by providing the necessary data. It is then utilized by Shenron to produce comprehensive 3D radar outputs, enhancing the fidelity of radar data in autonomous driving simulations.

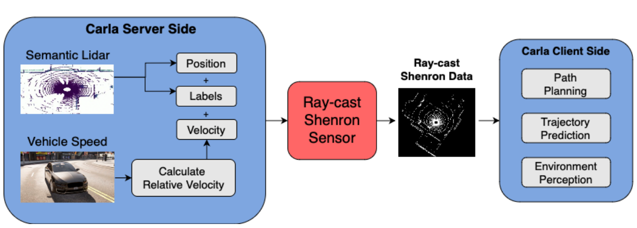


Figure 2. C-Shenron as the Shenron integration in CARLA

4.1. Challenges in integrating Shenron within CARLA

Sensors in CARLA follow a pipeline that transforms the raw sensor data into a usable format. Each sensor type is represented as a special actor within the simulation. The sensor actor interacts with the simulated environment and continuously gathers data based on its type and configuration.

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, traffic management, and sensor data generation. It also manages the communication with the client, transmitting sensor data and receiving control inputs from the client. The client application, typically written in Python, receives sensor data from the server, processes it, and sends control commands back to the server. Sensors in CARLA retrieve data either at every simulation step or when the specific event occurs. For example, the camera generates images at every frame, whereas collision sensors are activated upon detecting an event. The collected raw sensor data, along with metadata such as sensor type, frame number and timestamp, is serialized and transmitted to the client application via a real time communication protocol.

On the client side, applications can subscribe to a sensor's data stream. When a new data frame arrives, a registered callback function is triggered. This function deserializes the data stream back into a SensorData object and processes it further. This modular design allows for the integration of the custom sensors. However, the core sensor actor, data stream, and server to client communication are implemented in C++ using specific data structures and functions.

4.2. C-Shenron

To seamlessly integrate the Python-based Shenron sensor into the C++-based CARLA simulation environment, we devised a hybrid approach that addresses the fundamental challenges posed by this integration. We introduced a custom C++ Raycast Shenron sensor on the server side to capture point cloud data, including semantic segmentation and relative velocity information. This approach aligns with

CARLA's native C++ architecture, ensuring efficient communication and integration with the core simulation loop. The data collected by the Raycast Shenron sensor along with metadata is then transmitted to the client side. On the client side, Shenron processes the received data to generate the simulated radar data. To mitigate the real-time latency introduced by the Shenron processing, we paused the CARLA simulation during this phase, ensuring that the overall simulation time remains unaffected. The Figure 2 represents the overall picture of the Shenron integration with CARLA.

4.2.1. Relative Velocity Calculation

We implement the functions required to calculate the relative velocity in our new Raycast Shenron sensor. We compute the relative velocity v_{rel} of a detected target relative to the Raycast Shenron sensor, $\mathbf{v_s}$. It retrieves the target's velocity, $\mathbf{v_t}$, and calculates the normalized direction vector, $\mathbf{d}$, from the Raycast Shenron sensor to the target. By finding the velocity difference between the target and the sensor and taking the dot product with this direction vector, the function isolates the component of relative velocity along the line connecting the sensor and the target. This result, represents the target's velocity relative to the Raycast Shenron,

$$v_{rel} = (\mathbf{v_t} - \mathbf{v_s}) \cdot \mathbf{d}$$

$$\mathbf{d} = \frac{\mathbf{p_t} - \mathbf{p_s}}{\|\mathbf{p_t} - \mathbf{p_s}\|}$$

where $\mathbf{p_t}$ and $\mathbf{p_s}$ are the position vectors of the sensor and target respectively.

4.2.2. Dense Point Cloud Generation

To generate a dense point cloud data with a complete 360-degree field of view at each simulation step, we concatenated two 180-degree frames, aligning the previous frame with the current ego-vehicle position. By capturing two half-frames and combining them, we effectively doubled the point cloud density, resulting in a more accurate and detailed representation of the surrounding environment. This approach was crucial to generate realistic radar signals.

We developed a comprehensive solution that facilitates the integration of Shenron sensor into the CARLA system seamlessly. Additionally, we provide example scripts to simulate and visualize the Shenron radar data within the CARLA environment, demonstrating how to effectively use this radar in your simulations. Detailed instructions and resources are available as open source on the following GitHub repository: CARLA-Shenron-release.

5. Implementation

In this section we'll dive into how we utilized integrated Shenron in CARLA to train a end-to-end Perception and Driving model, built on top of the Transfuser++ architecture [16].

5.1. End-to-end driving with CARLA Garage

Safe navigation is the ultimate goal of a self-driving car, which includes identifying obstacles, planning the path around them and eventually reaching the goal. Integrating a realistic sensor model in CARLA gives us the ability to test the effect of radar algorithms on downstream tasks like path planning and navigation. Hence we use this opportunity to perform extensive experimentation on the effect of using radar data on downstream tasks. In this section we first describe the end-to-end driving system used for perception and planning followed by the results obtained when we evaluated navigation performance achieved by using radar.

5.1.1. CARLA Garage

We use the CARLA Garage [14] platform for generation of high-quality data and training of end-to-end autonomous driving models. The platform provides supports integration and deployment of both pretrained and custom models, offering necessary scripts and tools for dataset generation, model training, and benchmark evaluations, thus streamlining the process. Through this platform, we customized across multiple sensor placements and input data for training the end-to-end Deep Learning model of our choice. The output of the model is used as control signals for actions such as steering, brakes and gas that can be used to drive a autonomous agent in the simulation.

5.2. Dataset Generation

CARLA employs an expert autonomous agent that emulates driving of an experienced human driver, producing highly reliable driving data which is essential for training large autonomous driving models. This expert agent follows predefined traffic rules, navigates traffic scenarios, and interacts safely with obstacles just like an experienced human driver would perform. This expert also has direct access to detailed map data like lane boundaries, traffic signals, speed limits, and waypoints, enabling precise route planning and rule compliance without relying on raw sensor interpretation. Additionally, the expert bypasses complex object detection, directly retrieving the exact locations, velocities, and classifications of vehicles, pedestrians, and obstacles, thereby eliminating perception errors and ensuring reliable tracking. Furthermore, it has perfect localization within the environment, sidestepping common errors in real-world localization methods like GPS and LiDAR. This access to precise data enables the generation of a robust, high-quality dataset, ideal for training and benchmarking autonomous systems in controlled simulations.

To accelerate data collection, we launch multiple CARLA instances in parallel, allowing simultaneous data generation across various scenarios and weather conditions.

This approach enhances the dataset's diversity and richness, reducing the collection time from days to hours. Using a Kubernetes cluster, we launch 210 jobs, each corresponding to a distinct CARLA instance for different route-scenario combinations across all 8 CARLA towns (Town01-Town07 and Town10), reserving Town08 and Town09 for evaluation. This results in 70 unique combinations, with each combination repeated thrice, yielding a total of 555k frames. For our experiments, we only train the model on 185k frames, excluding repetitions. The additional data gathered may be utilized in future experiments to assess the impact of a larger training dataset on model performance. We will also release the complete collected dataset for the research community.

5.3. Integrating with TF++ Architecture

In CARLA Garage, we employ Transfuser++, a state-of-the-art model, for both perception and planning tasks. The Transfuser++ architecture features a transformer-based sensor fusion module that integrates camera and LiDAR data, alongside auxiliary branches for perception tasks like classification, detection, and segmentation. In our evaluations, we don't utilize any of these auxiliary branches and only use the Transformer encoders and decoders. Additionally, it includes a transformer decoder to output the target speed and path for the autonomous vehicle. For more details on the architecture, refer to [16]. In our implementation, we create high-fidelity radar data from Shenron as range-angle plots and input these images directly into the BEV branch, bypassing the LiDAR images as seen in Figure 3, and further conduct end-to-end training and evaluation of this model.

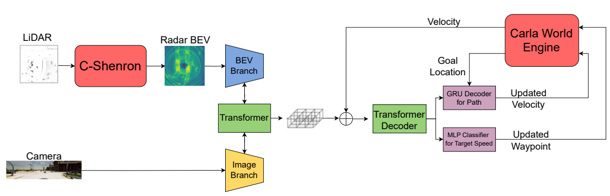


Figure 3. C-Shenron with the Transfuser++ Architecture

5.4. Training details

To train our model, we adopted the same loss function employed in the Transfuser++ architecture [16]. Our training process involved a batch size of 12 and 30 epochs. We utilized a learning rate of $3\text{x}10^{-4}$, and trained the model on a system equipped with 6 NVIDIA A10 GPUs, which required approximately 2 days to complete the training process.

6. Evaluation

In this section, we evaluate our trained model, which incorporates the Shenron sensor system, by comparing its driving performance with that of current state-of-the-art end-to-end driving models. Our primary model for processing autonomous vehicle sensor data is Transfuser++ [16]. We also present two case studies that explore varying radar sensor placements and assess the impact of these configurations. Our results indicate that radar images can serve as an effective alternative to LiDAR, delivering comparable performance along with enhanced all-weather capability. These results reopen the field for utilizing radars in end-to-end autonomous driving.

6.1. Metrics

The driving proficiency of an autonomous agent is evaluated through various metrics provided by CARLA that gives insights into different aspects of driving behavior. In the context of our setup, we evaluate on a set of metrics that offers a comprehensive understanding of the agent's performance. The specific metrics are :-

- **Driving Score**: The primary metric of the leaderboard, calculated as the product of the other two metrics: route completion and the infractions penalty.
- **Route completion**: It is the percentage of the route distance completed by an agent.
- **Infraction Penalty**: The leaderboard tracks multiple types of infractions, and this metric consolidates all infractions triggered by an agent into a single score, calculated as a geometric series.

In the CARLA simulation, infractions are penalized based on severity. For example, collisions with pedestrians, vehicles, and static objects incur varying penalties. Traffic violations, such as running red lights or stop signs, also result in higher penalties. Indefinite blockage of the vehicle leads to a timeout and additional penalties.

Agents must adhere to surrounding traffic speeds and yield to emergency vehicles, with noncompliance resulting in further penalties. Driving off-road negatively affects the route score, as that segment is excluded. Certain events, like significant deviations from the route or prolonged inactivity, can lead to a simulation shutdown. Each of these incidents is meticulously recorded, providing comprehensive insights into the performance of the agent throughout the simulation [23]. Once all routes are completed, an overall metric for each of the three types is calculated by taking the arithmetic mean of all individual route metrics combined.

6.2. Case Studies

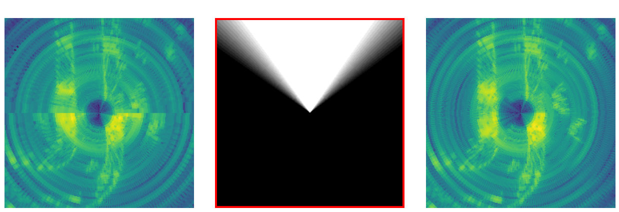
We evaluate our models using the routes from NEAT [7] paper, which include various settings like highways, urban areas, and residential zones with diverse road layouts and obstacles to simulate urban conditions. Agents face traffic scenarios based on NHTSA typology, such as navigating intersections, responding to pedestrians, cyclists, and other road users, and many more. To ensure consistency, each model was tested on the same set of 14 routes over 5 iterations under stable, moderate conditions without extreme weather. Additionally, we carried out two case studies to examine the impact of different sensor placements and the impact of each radar view on performance in end-to-end driving tasks.

6.2.1. Does increasing radar views help?

In this case study we analyze the potential benefits 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 and back radars, and full coverage with front, back, left, and right radars (we will denote as FBLR). All configurations are also fused with camera features. Note that all the views of radar have 180° FOV.

When using front and back radar views, combining them is straightforward; the two can simply be concatenated vertically to create a complete 360° image, as illustrated in Figure 4a. However, an interesting challenge arises when attempting to merge the four radar views into a single high-quality image. A basic method would be to extract 90° FOV from each image and arrange them in a circular pattern, but this approach is inefficient. Shenron-generated radar images contain concentric circular lines with slightly varying radii, depending on the view, resulting in diagonal lines and irregular patterns across the combined radar image, which impairs perception. This can also be seen in Figure 4a, where a horizontal line is present in the middle of the image.

An alternative approach involves overlapping of border regions from different views to average out this inconsistency. This technique uses a specialized mask as seen in Figure 4b which are then rotated for proper orientation and 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 composite radar image Figure 4c demonstrates the efficacy of this approach and creates an accurate representation of the vehicle's surroundings.



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

Figure 4. Images representing: (a) Radar image after Front+Back concatenation, (b) Mask for FBLR concatenation, (c) Radar image after FBLR concatenation.

The findings are presented in Table 2. 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. Among the radar models, the Front+Back configuration demonstrates the best performance across all metrics, significantly outperforming the LiDAR model (+6 in DS and +0.05 in IS). The Front-only radar model also surpasses LiDAR, indicating that a single radar view can exceed the baseline performance. Notably, the FBLR configuration has a lower DS than Front+Back, likely due to its low RC score; however, it exhibits more stability and lower variance, suggesting that additional field-of-view sensors enhance consistency.

Radar View	DS ↑	RC ↑	IS ↑
LiDAR (ours)	76.84 ± 5.26	95.93 ± 3.43	0.79 ± 0.05
Front	79.97 ± 5.36	96.52 ± 3.02	0.82 ± 0.06
Front+Back	**82.39 ± 4.87**	**97.03 ± 2.95**	**0.84 ± 0.03**
FBLR	79.24 ± 1.85	93.56 ± 2.75	0.84 ± 0.05
Expert	93.82	97.394	0.964

Table 2. Results for different radar views with Driving Score (DS), Route Completion (RC) and Infraction Score (IS).

Lastly, the Expert model represents statistics from CARLA's driver agent, which sets a theoretical upper performance limit as its training data was derived from this agent. Although none of the models achieve expert performance, the Front+Back radar configuration is the closest across all metrics. Overall, radar-based models outperforms the Camera + LiDAR setups in all key areas.

Looking deeper into driving scores from Figure 5, in Urban routes, Front+Back performed slightly better than FBLR, suggesting rear radar coverage is beneficial in congested traffic. On Highways, FBLR greatly outperformed the others, which outlines the importance of 360-degree radar for detecting vehicles from multiple directions. Overall, additional radar views enhance performance across routes, with the greatest impact on highways.

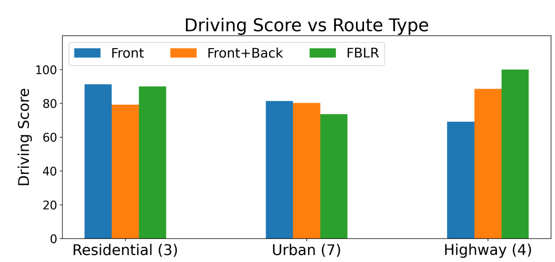


Figure 5. Route-wise Driving Score for Multiple Radar Views. The three categories have 3, 7 and 4 routes respectively.

Table 3 presents other scores where the FBLR configuration excels in detecting vehicular collisions, static objects, and route deviations, while the Front+Back configuration performs best in red light infractions and agent timeouts. FBLR's strong vehicle detection benefits from its multi-directional radar views, enhancing its ability to avoid obstacles (perfect score in static object detection). However, the Front+Back configuration minimizes red light infractions and completely avoids timeouts, suggesting that fewer inputs simplify decision-making. In conclusion, FBLR is optimal for environmental awareness, while Front+Back excels in rapid decision-making situations, both surpassing driving performances by LiDAR.

Radar View	Veh ↓	Stat ↓	Red ↓	Dev ↓	TO ↓
LiDAR	0.62 ± 0.16	**0.00**	0.14 ± 0.09	0.02 ± 0.05	0.04 ± 0.04
Front	0.51 ± 0.21	0.06 ± 0.04	0.05 ± 0.06	0.06 ± 0.04	**0.00**
Front+Back	0.43 ± 0.12	0.01 ± 0.02	**0.05 ± 0.04**	0.01 ± 0.03	**0.00**
FBLR	**0.32 ± 0.06**	**0.00**	0.26 ± 0.10	**0.00**	0.09 ± 0.08
Expert	0.00	0.00	0.00	0.00	0.14

Table 3. Results for different radar views with Vehicle Infractions (Veh), Static Object Collisions (Stat), Red Light Infractions (Red), Route Deviations (Dev) and Agent Time Outs (TO).

6.2.2. Redaction of Radar views

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

Redact	DS ↑	RC ↑	IS ↑
Camera only	64.35	85.83	0.70
Left	75.79 ± 1.79	93.65 ± 2.68	0.78 ± 0.02
Right	**76.61 ± 3.00**	91.06 ± 0.91	**0.82 ± 0.04**
Front	35.88 ± 8.63	91.07 ± 3.66	0.37 ± 0.10
Back	73.30 ± 4.25	**96.16 ± 3.80**	0.77 ± 0.03
No Redact	79.24 ± 1.85	93.56 ± 2.75	0.84 ± 0.03

Table 4. Redaction of radar results with Driving Score (DS), Route Completion (RC) and Infraction Score (IS).

The results from the Table 4 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 as well. The camera only model performs the least across all the scores indicating that having radar views helps the model.

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

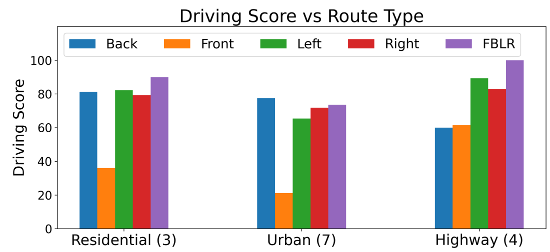


Figure 6. Route-wise Driving Score for Redaction of Radar. The three categories have 3, 7 and 4 routes respectively.

Redact	Veh ↓	Stat ↓	Red ↓	Dev ↓	TO ↓
Camera only	1.661	0.73	0.00	0.00	0.14
Left	0.60 ± 0.14	**0.00**	0.37 ± 0.08	0.19 ± 0.02	0.17
Right	**0.47 ± 0.22**	0.02 ± 0.04	0.21 ± 0.06	**0.20 ± 0.05**	0.14 ± 0.07
Front	4.60 ± 0.76	0.42 ± 0.15	0.64 ± 0.20	0.46 ± 0.12	0.24 ± 0.14
Back	0.87 ± 0.18	**0.00**	**0.11 ± 0.07**	0.30 ± 0.13	**0.11 ± 0.06**
No Redact	0.32 ± 0.06	0.00	0.26 ± 0.10	0.00	0.09 ± 0.08

Table 5. Redaction of radar results with Vehicle Infractions (Veh), Static Object Collisions (Stat), Red Light Infractions (Red), Route Deviations (Dev) and Agent Time Outs (TO).

Similar results are observed from Table 5 where in the front radar leads to an unusually high vehicle detection score, likely due to misclassification. Overall, the model performs best with all views present, showcasing that each radar view offers unique contributions, with the front view essential for vehicle detection and stability.

7. Future Work

In future work, we aim to extend our evaluation of C-Shenron in CARLA by incorporating a more diverse set of routes from NEAT and other evaluations. Furthermore, we plan to add effective fusion techniques from multiple views of radars. We would also like to evaluate more low and high resolution radars. Additionally, longer and more varied routes will be incorporated to demonstrate the robustness of all community approaches. Expanding the evaluation to cover a broader range of towns and conditions will also allow for more comparisons between our radar-based model and other state-of-the-art (SOTA) models beyond Transfuser++.

References

[1] Prashant Ahire, SMM Naidu, Sandeep Varpe, Sakshi Nadarge, and Anushree Patil. Simulating vehicle driving using carla. *Journal of Electrical Systems*, 20(10s):44–52, 2024.

[2] Mina Alibeigi, William Ljungbergh, Adam Tonderski, Georg Hess, Adam Lilja, Carl Lindström, Daria Motorniuk, Junsheng Fu, Jenny Widahl, and Christoffer Petersson. Zenseact open dataset: A large-scale and diverse multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF*

International Conference on Computer Vision, pages 20178–20188, 2023.

[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*, pages 340–353, 2020.

[4] Kshitiz Bansal, Gautham Reddy, and Dinesh Bharadia. Shenron - scalable, high fidelity and efficient radar simulation. *IEEE Robotics and Automation Letters*, 9(2):1644–1651, 2024.

[5] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9297–9307, 2019.

[6] Dian Chen, Brady Zhou, Vladlen Koltun, and Philipp Krähenbühl. Learning by cheating. In *Conference on Robot Learning*, pages 66–75. PMLR, 2020.

[7] 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*, pages 15793–15803, 2021.

[8] 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*, 45(11):12878–12895, 2022.

[9] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[10] CARLA Simulator Developers. Radar sensor model limitations in carla simulator. `https://github.com/carla-simulator/carla/issues/4974`, 2024.

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

[12] Divya Garikapati and Sneha Sudhir Shetiya. Autonomous vehicles: Evolution of artificial intelligence and the current industry landscape. *Big Data and Cognitive Computing*, 8 (4):42, 2024.

[13] 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*, pages 3354–3361. IEEE, 2012.

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

[15] Anthony Hu, Gianluca Corrado, Nicolas Griffiths, Zachary Murez, Corina Gurau, Hudson Yeo, Alex Kendall, Roberto Cipolla, and Jamie Shotton. Model-based imitation learning for urban driving. *Advances in Neural Information Processing Systems*, 35:20703–20716, 2022.

[16] 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.

[17] Li Li, Khalid N Ismail, Hubert PH Shum, and Toby P Breckon. Durlar: A high-fidelity 128-channel lidar dataset with panoramic ambient and reflectivity imagery for multi-modal autonomous driving applications. In *2021 International Conference on 3D Vision (3DV)*, pages 1227–1237. IEEE, 2021.

[18] Fei Liu, Zihao Lu, and Xianke Lin. Vision-based environmental perception for autonomous driving. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, page 09544070231203059, 2022.

[19] Eshed Ohn-Bar, Aditya Prakash, Aseem Behl, Kashyap Chitta, and Andreas Geiger. Learning situational driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11296–11305, 2020.

[20] Quang-Hieu Pham, Pierre Sevestre, Ramanpreet Singh Pahwa, Huijing Zhan, Chun Ho Pang, Yuda Chen, Armin Mustafa, Vijay Chandrasekhar, and Jie Lin. A* 3d dataset: Towards autonomous driving in challenging environments. In *2020 IEEE International conference on Robotics and Automation (ICRA)*, pages 2267–2273. IEEE, 2020.

[21] Arvind Srivastav and Soumyajit Mandal. Radars for autonomous driving: A review of deep learning methods and challenges. *IEEE Access*, 2023.

[22] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[23] CARLA Simulator Team. Carla autonomous driving leaderboard. `https://leaderboard.carla.org/`, 2024.

[24] Marin Toromanoff, Emilie Wirbel, and Fabien Moutarde. End-to-end model-free reinforcement learning for urban driving using implicit affordances. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7153–7162, 2020.

[25] Shanliang Yao, Runwei Guan, Xiaoyu Huang, Zhuoxiao Li, Xiangyu Sha, Yong Yue, Eng Gee Lim, Hyungjoon Seo, Ka Lok Man, Xiaohui Zhu, et al. Radar-camera fusion for object detection and semantic segmentation in autonomous driving: A comprehensive review. *IEEE Transactions on Intelligent Vehicles*, 2023.