Editable Scene Simulation for Autonomous Driving via Collaborative LLM-Agents

Yuxi Wei[1,5*] Zi Wang[3,5*] Yifan Lu[1,5*] Chenxin Xu[1,5*]
Changxing Liu[1,5] Hao Zhao[4] Siheng Chen[1,2,5†] Yanfeng Wang[1,2]

[1] Shanghai Jiao Tong University [2] Shanghai AI Laboratory
[3] Carnegie Mellon University [4] Tsinghua University [5] Multi-Agent Governance & Intelligence Crew (MAGIC)

{wyx3590236732, yifan_lu, xcxwakaka, cx-liu}@sjtu.edu.cn
{sihengc, wangyanfeng}@sjtu.edu.cn
ziwang2@andrew.cmu.edu zhaohao@air.tsinghua.edu.cn

Abstract

Scene simulation in autonomous driving has gained significant attention because of its huge potential for generating customized data. However, existing editable scene simulation approaches face limitations in terms of user interaction efficiency, multi-camera photo-realistic rendering and external digital assets integration. To address these challenges, this paper introduces ChatSim, the first system that enables editable photo-realistic 3D driving scene simulations via natural language commands with external digital assets. To enable editing with high command flexibility, ChatSim leverages a large language model (LLM) agent collaboration framework. To generate photo-realistic outcomes, ChatSim employs a novel multi-camera neural radiance field method. Furthermore, to unleash the potential of extensive high-quality digital assets, ChatSim employs a novel multi-camera lighting estimation method to achieve scene-consistent assets' rendering. Our experiments on Waymo Open Dataset demonstrate that ChatSim can handle complex language commands and generate corresponding photo-realistic scene videos. Code can be accessed at: https://github.com/yifanlu0227/ChatSim.

1. Introduction

Perception [12, 15–17, 38, 66] is the window of an autonomous vehicle into the external environment. To ensure the robustness of the vehicle's perceptual capabilities during both training and testing phases, it necessitates the collection of high-quality perception data in substantial volumes [13, 58]. However, the operation of a fleet for the acquisition of real-world data often incurs prohibitive expenses, particularly for specialized or customized requirements. For instance, in the aftermath of an accident or intervention involving an autonomous vehicle, it is imperative to test the vehicle's perception system across a spectrum of similar scenarios. While replicating such scenario data from real-world instances is nearly impossible due to the uncontrollability of actual scenes [9, 52], customized scene simulation emerges as a vital and feasible alternative. It enables the precise modeling of specific conditions without high costs and logistical complexities of real-world data collection [4, 72].

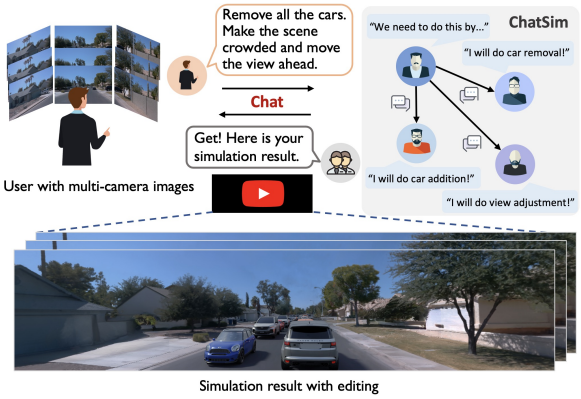


Figure 1. *ChatSim* enables the editing of photo-realistic 3D driving scene simulations via language commands.

To effectively simulate customized driving scenes, we identify three key properties as fundamental. First, the simulation should be capable of following sophisticated or abstract demands, thereby facilitating the production. Second, the simulation should generate photo-realistic, view-consistent outcomes, which allow for the closest approximation to vehicle observations in real-world scenarios. Third, it should allow for the integration of external digital assets [6, 44] with their photo-realistic textures and materials while fitting the lighting conditions. This capability would unlock the potential for data expansion by incorporating a wide array of external digital assets, satisfying customized needs.

A vast array of significant works have been proposed for scene simulation, yet they fail to meet all three of

*Equal contribution. †Corresponding author.

these requirements. Traditional graphics engines, such as CARLA [22] and UE [25], offer editable virtual environments with external digital assets, but the data realism is restricted by asset modeling and rendering qualities. Image generation based methods, such as BEVControl [71], DriveDreamer [63], MagicDrive [26], can generate realistic scene images based on various control signals, including BEV maps, bounding boxes and camera poses. However, they struggle to maintain view consistency and face challenges in importing external digital assets due to the absence of 3D spatial modeling. Rendering-based methods have been proposed to obtain photo-realistic and view-consistent scene simulation. Notable examples like UniSim [73] and MARS [68] come equipped with a suite of scene-editing tools. However, these systems require extensive user involvement in every trivial editing step via code implementation, which is ineffective when performing the editing. Furthermore, while they handle vehicles in observed scenarios effectively, their inability to support external digital assets restricts opportunities for data expansion and customization.

To fulfill the identified requirements, we introduce *ChatSim*, the first system that enables editable photo-realistic 3D driving scene simulations via natural language commands with external digital assets. To use *ChatSim*, users simply engage in a conversation with the system, issuing commands through natural language without any involvement in intermediate simulation steps; see Figure 1 for illustration.

To address complex or abstract user commands effectively, *ChatSim* adopts a large language model (LLM)-based multi-agent collaboration framework. The key idea is to exploit multiple LLM agents, each with a specialized role, to decouple an overall simulation demand into specific editing tasks, thereby mirroring the task division and execution typically founded in the workflow of a human-operated company. This workflow offers two key advantages for scene simulation. First, LLM agents' ability to process human language commands allows for intuitive and dynamic editing of complex driving scenes, enabling precise adjustments and feedback. Second, the collaboration framework enhances simulation efficiency and accuracy by distributing specific editing tasks among specialized agents, ensuring detailed and realistic simulations with improved task completion rates.

To generate photo-realistic outcomes, we propose McNeRF in *ChatSim*, a novel neural radiance field method that incorporates multi-camera inputs, offering a broader scene rendering. This integration fully exploits camera setups on vehicles but raises two significant challenges: camera pose misalignment due to asynchronized trigger times and brightness inconsistency due to different camera exposure times. To address camera pose misalignment, McNeRF uses a multi-camera alignment to reduce extrinsic parameter noises, ensuring rendering quality. To address brightness inconsistency, McNeRF integrates the critical exposure times to recover scene radiance in HDR, markedly mitigating the issue of color discrepancies at the intersections of two camera images with different exposure times.

To import external digital assets with their realistic textures and materials, we propose McLight, a novel multi-camera lighting estimation that blends skydome and surrounding lighting. Our skydome estimation restores accurate sun behavior with peak intensity residual connection, enabling the rendering of prominent shadows. For surrounding lighting, McLight queries McNeRF to achieve complex location-specific illumination effects, like those in the tree shade with sunlight being blocked. This significantly improves the rendering realism of the integrated 3D assets.

We conduct extensive experiments on the Waymo autonomous driving dataset and show that *ChatSim* generates photo-realistic customized perception data including dangerous corner cases according to various human language commands. Our method is compatible with mixed, highly-abstract and multi-round commands. Our method achieves SoTA performance with an improvement of 4.5% in photo-realism with a wide-angle rendering. Moreover, we demonstrate our lighting estimation outperforms the SoTA methods both qualitatively and quantitatively, reducing the intensity error and angular error by 57.0% and 9.9%.

2. Related Work

Scene simulation for autonomous driving. Current scene simulation methods can be generally divided into three categories: graphics engines, image generation, and scene rendering. Graphics engines, such as CARLA [22], AirSim [54], OpenScenario Editor [24], 51Sim-One [1] and RoadRunner [21], create a virtual world for simulating a wide range of driving scenarios. However, there exists a significant domain gap between the virtual world and reality. Image generation methods can generate realistic scene images based on different control signals, such as HD maps [26, 39, 59], sketch layout [71], bounding boxes [26, 39, 63], text [26, 33, 39, 63] and driving actions [33, 63]. However, these approaches can hardly maintain scene consistency. To obtain a coherent driving scene, methods based on scene rendering target to reconstruct the 3D scene. READ [41] employs point clouds and uses a U-Net to render images. With the rapid development of Neural Radiance Field (NeRF) [7, 8, 43, 45, 56, 62], several works [28, 34, 47, 48, 61, 68, 69, 73] also exploit NeRFs to model cars and static street backgrounds in outdoor environments. Moreover, notable examples like UniSim [73] and MARS [68] come equipped with a suite of scene-editing tools. However, these methods require extensive user involvement in intermediate editing steps and they fail to support external digital assets for data expansion. In this work, we propose *ChatSim* that achieves automatic simulation editing via language commands and integrates external digital assets to enhance realism and flexibility. In *ChatSim*,

Method	Photo-realistic	Dim.	Multi-camera	Editable	External assets	Language	Open-source
CARLA [22]	×	3D	✓	✓	✓	×	✓
AirSim [54]	×	3D	✓	✓	✓	×	✓
OpenScenario [24]	×	3D	✓	✓	✓	×	✓
51Sim-One [1]	×	3D	✓	✓	✓	×	×
RoadRunner [21]	×	3D	✓	✓	✓	×	✓
BEVGen [59]	✓	2D	✓	✓	×	×	✓
BEVControl [71]	✓	2D	✓	✓	×	×	×
DriveDreamer [63]	✓	2D	✓	✓	×	✓	×
DrivingDiffusion [39]	✓	2D	✓	✓	×	✓	×
GAIA-1 [33]	✓	2D	×	✓	×	✓	×
MagicDrive [26]	✓	2D	✓	✓	×	×	×
READ [41]	✓	3D	×	×	×	×	✓
Neural SG [47]	✓	3D	×	✓	×	×	✓
Neural PLF [48]	✓	3D	×	×	×	×	✓
S-NeRF [69]	✓	3D	✓	×	×	×	✓
UniSim [73]	✓	3D	×	✓	×	×	×
MARS [68]	✓	3D	×	✓	×	×	✓
ChatSim (Ours)	✓	3D	✓	✓	✓	✓	✓

Table 1. Comparison of existing and proposed methods for autonomous driving simulation.

we integrate McNeRF, a novel neural radiance field designed to leverage multi-camera inputs for high-fidelity rendering.

Lighting estimation. Lighting estimation focuses on assessing the illumination conditions of a real-world environment to seamlessly integrate digital objects. Early methods [35, 36] for outdoor environments use explicit cues like detected shadows on the ground. Recent works usually adopt learning-based approaches [27, 30, 31, 37, 40, 74] by predicting different lighting representations like spherical lobes [10, 40], light probes [37], environment map [53, 55], HDR sky model [31, 64, 74] and lighting volume [64]. However, few of them consider multi-camera input, which is common for driving scenarios. In this paper, we propose a novel multi-camera lighting estimation method, McLight, combining with our McNeRF, to estimate a wider range of lighting and obtain the spatially-varying lighting effects of assets.

Large language model and collaborative framework. Large Language Models (LLMs) are AI systems trained on extensive data to understand, generate, and respond to human language. GPT [11] is a pioneering work to generate human-like content. The following updated versions GPT-3.5 [14] and GPT-4 [46], provide more intelligent capabilities like chatting, browsing and coding. Notable other large language models include InstructGPT [49], LLaMA [60] and PaLM [5, 18]. Based on LLM, many works [3, 23, 29, 65, 76] improve the problem-solving abilities by integrating communication among multiple agents. [32] and [67] define a group of well-organized agents to form operating procedures with conversation and code programming. In this paper, we exploit the power of collaborative LLM agents in simulation for autonomous driving, enabling the various editing of 3D scenes via language commands.

3. Collaborative LLM-Agents for Editing

The *ChatSim* system analyzes specific user commands and returns a video that meets customized needs; see Figure 2. Since user commands could be abstract and sophisticated, it requires the system to have flexible task-handling ability. Directly applying a single LLM agent struggles with multi-step reasoning and cross-referencing. To address this, we design a series of collaborative LLM agents, where each agent is responsible for a unique aspect of the editing task.

3.1. Specific Agent's Functionality

Agents in *ChatSim* comprise two key components: a Large Language Model (LLM) and the corresponding role functions. The LLM is responsible for understanding the received commands while the role functions process the received data. Each agent is equipped with unique LLM prompts and role functions tailored to their specific duties within the system. To accomplish their tasks, agents first convert the received commands to a structured configuration using LLM with the assistance of prompts. Then the role functions utilize the structured configuration as parameters to process the received data and produce the desired outcomes; see an agent example on the right side of Figure 2. This workflow endows agents with both language interpretation capabilities and precise execution capabilities.

Project Manager Agent. The project manager agent decomposes direct commands into clear natural language instructions dispatched to other editing agents. To equip the project manager agent with the capability of command decomposition, we design a series of prompts for its LLM. The core idea of the prompts is to describe the action set, give the overall goal, and define the output form with examples; The role functions send the decomposed instructions to other agents for editing. The presence of the project manager agent enhances the system's robustness in interpreting various inputs and streamlines operations for clarity and fine granularity.

Tech agent for view adjustment. The view adjustment agent generates suitable extrinsic camera parameters. The LLM in the agent translates the natural language instructions for viewpoint adjustment into movement parameters to the target viewpoint's position and angle. In role functions, these movement parameters are turned into transformation matrices required by the extrinsic, which are then multiplied by the original parameters to yield a new viewpoint.

Tech agent for background rendering. The background rendering agent renders the scene background based on multi-camera images. The LLM receives the rendering command and then operates the role functions for rendering. Notably, in role functions, we specifically integrate a novel neural radiance field method (McNeRF) taking multi-camera inputs and considering exposure time, solving the problem of blurring and brightness inconsistency in multi-camera rendering, see more details in Section 4.1.

Tech agent for vehicle deleting. The vehicle deleting agent removes specified vehicles from the background. It first identifies current vehicle attributes like 3D bounding boxes and colors from given scene information or results from a scene

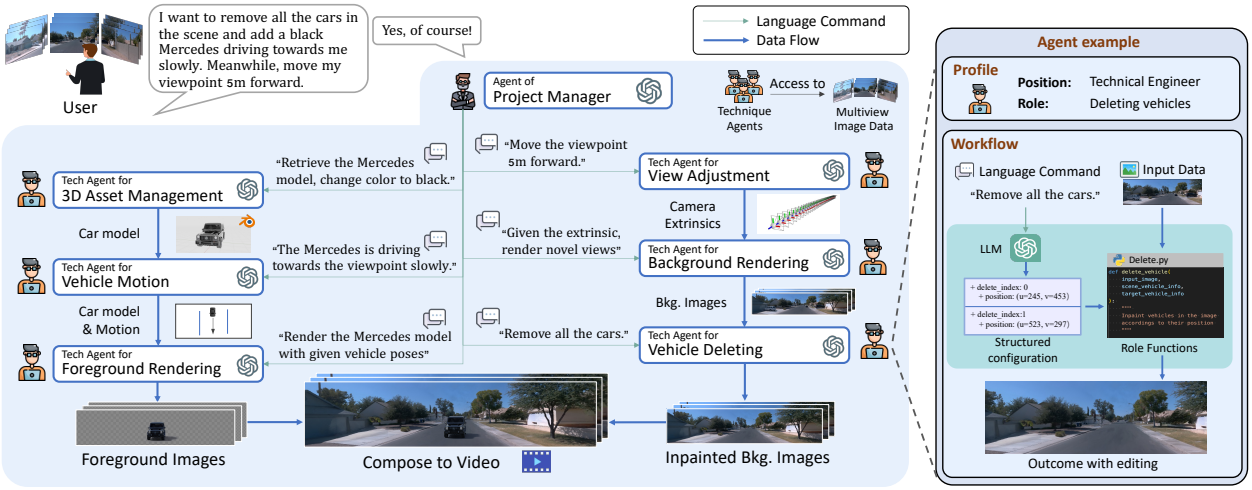


Figure 2. ***ChatSim* system overview.** The system exploit multiple collaborative LLM agents with specialized roles to decouple an overall demand into specific editing tasks. Each agent equips an LLM and corresponding role functions to interpret and execute its specific tasks.

```
# General task description
"I will give you a transformation operation for my viewpoint, which may include translation
in 'x', 'y', 'z' or a rotation 'theta' around z-axis. "
# Interpretation of details
"For translation, positive 'x' represents forward, positive 'y' represents left, and 'z'
represents up. It follows a left-hand coordinate system."
"For rotation, positive 'theta' is counterclockwise. So from own perspective, my viewpoint
turns to the left. 'theta' is in degree."
# Return format
"Given my operation, return a dictionary in JSON format, with keys 'x', 'y', 'z', 'theta'."
# Few-shot examples
"I will give you some examples:
<user>: rotate the viewpoint 30 degrees to the left; <assistant>: {\n  'x': 0,\n  'y': 0,\n
'z': 0,\n  'theta': 30,\n } \
<user>: move the viewpoint to the right by 1; <assistant>: {\n  'x': 0,\n  'y': -1,\n
'z': 0,\n  'theta': 0,\n} "
```

Figure 3. Prompt example of view adjustment agent.

perception model like [42]. The LLM gathers attributes of the vehicles and performs matching with user requests. Upon confirming the targeted vehicles, it employs a per-frame inpainting model as the role functions, such as latent diffusion methods [51], to effectively delete them from the scene.

Tech agent for 3D asset management. The 3D asset management agent selects and modifies 3D digital assets according to user specifications. It constructs and maintains a 3D digital asset bank; see our bank details in the Appendix. To facilitate the addition of various objects, the agent first uses LLM to select the most suitable asset by key attributes matching with the requirements, such as color and type. If the matching is not perfect, the agent could modify the asset through its role functions like changing the color.

Tech agent for vehicle motion. The vehicle motion agent creates the initial places and subsequent motions of vehicles following the requests. Existing vehicle motion generation methods cannot directly generate motion purely from text and the scene map. To solve the problem, here we propose a novel text-to-motion method. The key idea is linking a placement and planning module as role functions with LLMs to extract and turn motion attributes into coordinates. Motion attributes include position attributes (e.g., distance, direction) and movement attributes (e.g., speed, action). For the placement module, we endow each lane node in the lane map with its attributes to match with the position attributes. The planning module plans the vehicle's approximate destination lane node and then plans the intermediate trajectory by fitting the Bezier curves. We also add trajectory tracking [70] to fit vehicle dynamics; see more details in the Appendix.

Tech agent for foreground rendering. The foreground rendering agent integrates camera extrinsic infomation, 3D assets, and motion information to render foreground objects in the scene. Notably, to seamlessly integrate the external assets with the current scene, we design a multi-camera lighting estimation method (McLight) into the role functions, coupling with McNeRF. The estimated illumination is then utilized by Blender API to generate foreground images. The detailed technical aspects will be elaborated in Section 4.2.

3.2. Agent Collaboration Workflow

Agents with tailored functions collaboratively work together to edit based on user commands. The project manager orchestrates and dispatches instructions to editing agents. The editing agents form two teams: background generation and foreground generation. For background generation, the background rendering agent generates rendered images using the extrinsic parameters from the view adjustment agent, followed by inpainting by the vehicle deleting agent. For foreground generation, the foreground rendering agent renders the images using the extrinsic parameters from the view adjustment agent, selected 3D assets from 3D asset management agent, and generated motions from vehicle motion agent. Finally, the foreground and background images are composed to create and deliver a video to the user. The editing information in each agent's configuration is recorded by the project manager agent for possible multi-round editings.

4. Novel Rendering Methods

Based on the collaborative LLM agents framework introduced in Section 3, this section presents two novel rendering

techniques to enhance photo-realism in simulations. To tackle the rendering challenges caused by multiple cameras, we propose multi-camera neural radiance field (McNeRF), a novel NeRF model considering the varied camera exposure times for visual consistency. To render realistic external digital assets with location-specific lighting and accurate shadows, we propose McLight, a novel hybrid lighting estimation method combined with our McNeRF. Note that McNeRF and McLight are leveraged by the background rendering agent and the foreground rendering agent, respectively.

4.1. McNeRF for Background Rendering

An autonomous vehicle typically equips multiple cameras to achieve a comprehensive perception view. However, this poses challenges for NeRF training due to the misaligned multi-camera poses from asynchronized camera trigger times and the brightness inconsistency originating from different exposure times. To address these challenges, the proposed McNeRF uses two techniques: multi-camera alignment and brightness-consistent rendering.

Multi-camera alignment. Autonomous vehicles, despite having a localization module for accurate camera poses, face challenges with asynchronous trigger times across multiple cameras. To align camera extrinsics for NeRF training, our core idea is to leverage a consistent spatial coordinate system provided by Agisoft Metashape [2] to align the images captured by multiple cameras at different timestamps.

Specifically, let $\mathcal{I}^{(i,k)}$ and $\xi^{(i,k)}$ be the image captured by the ith camera at the kth trigger and the corresponding camera pose in the vehicle's global coordinate space, respectively. We first input all images into Metashape for recalibration. The aligned camera pose is then obtained as:

$$\widehat{\xi}^{(i,k)} = T_{M\to G} \cdot \xi_M^{(i,k)},$$

where $\xi_M^{(i,k)}$ denotes the recalibrated camera pose in the Metashape's unified spatial coordinate space, and $T_{M\to G}$ is the transformation from the Metashape's coordinate space to the vehicle's global coordinate space. After alignment, the pose noise can be significantly reduced. Then, the aligned camera pose $\widehat{\xi}^{(i,t)}$ can be used to generate the origins and directions of rays for McNeRF, enabling high-fidelity rendering. The aligned pose can also facilitate the foreground rendering agent's operations.

Brightness-consistent rendering. The exposure times of cameras can differ substantially, causing significant brightness differences across images, hindering the NeRF training. As shown in Figure 4, McNeRF, addresses this by incorporating exposure times into HDR radiance fields, prompting brightness consistency.

We adopt F2-NeRF [62] as our backbone model to handle the unbounded scene, sampling K points along the ray $\mathbf{r}$ and estimating each point's HDR radiance $\mathbf{e}_k$ and density σ_k.

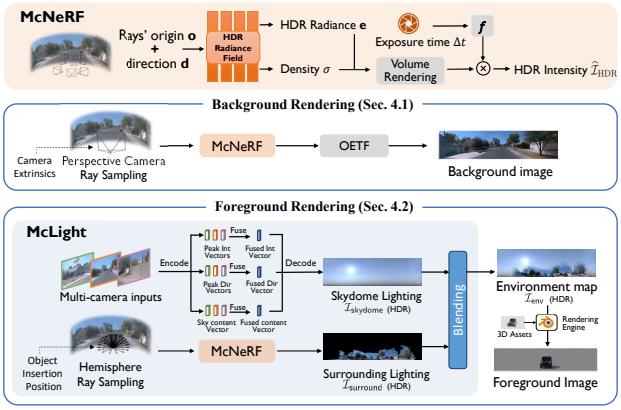


Figure 4. Rendering framework. The main components include McNeRF and McLight. Background rendering uses McNeRF to predict HDR pixel value and convert it to LDR with sRGB OETF. McLight includes a skydome lighting estimation network and adopts McNeRF to generate surrounding lighting.

The HDR light intensity is then calculated as:

$$\widehat{\mathcal{I}}_{\mathrm{HDR}}(\mathbf{r}) = f(\Delta t) \cdot \sum_{k=1}^{K} T_k \alpha_k \mathbf{e}_k, \tag{1}$$

where $\alpha_k = 1 - \exp(-\sigma_k \delta_i)$ is the opacity, δ_i is the point sampling interval, $T_k = \prod_{i=0}^{k-1}(1-\alpha_i)$ is the accumulated transmittance and Δt is the exposure time. The normalization function $f(\Delta t) = 1 + \epsilon(\Delta t - \mu)/\sigma$ is designed to stabilize training, where ϵ is a hyperparameter for scaling, μ and σ are the mean and standard deviation of the exposure times of all images, respectively.

By predicting scene radiance in HDR and multiplying it by the exposure time, we recover the light intensity received by the sensor and tackle the inconsistent color supervision at the intersections of two camera images with distinct exposure times. Moreover, the HDR light intensity outputted by McNeRF can provide scene-level illumination for foreground object rendering, a topic further discussed in Section 4.2.

To train the rendering network, we enforce the consistency of radiance between the rendered image (prediction) and the captured image (ground-truth). Given the ground-truth image $\mathcal{I}$, the loss function is then:

$$\mathcal{L} = \frac{1}{|R|} \sum_{\mathbf{r}\in R} \left(\mathrm{OETF}\left(\widehat{\mathcal{I}}_{\mathrm{HDR}}(\mathbf{r})\right) - \mathcal{I}(\mathbf{r}) \right)^2,$$

where R represents the ray set and $\mathrm{OETF}(\cdot)$ is the sRGB opto-electronic transfer function (gamma correction) [19] that converts HDR light intensity to LDR colors.

4.2. McLight for Foreground Rendering

To enrich the scene's content with substantial digital 3D assets, we employ Blender [20] foreground virtual objects' rendering. A seamless insertion critically depends on accurately estimating the scene's illumination conditions. Thus, as shown in Figure 4, we propose McLight, a novel hybrid lighting estimation consisting of skydome lighting and sur-

rounding lighting.

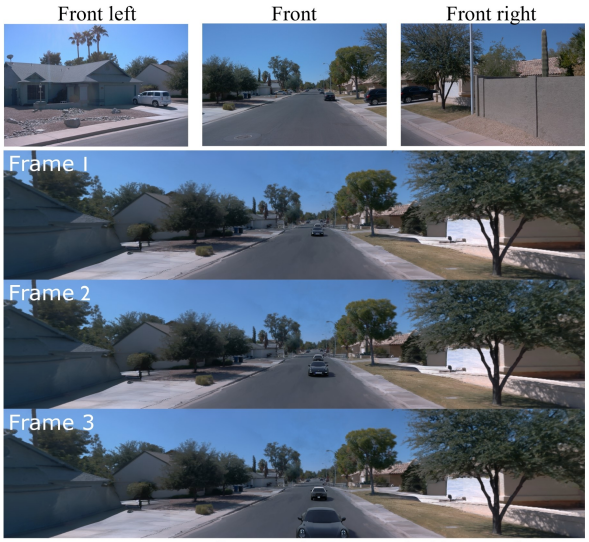
Skydome lighting estimation. Estimating skydome lighting from images is challenging for restoring accurate sun behavior. To achieve this, we propose a novel residual connection from the estimated peak intensity to the HDR reconstruction to address over-smoothing output. Further, we adopt a self-attention mechanism to fuse multi-camera inputs, capturing complementary visual cues.

Here we employ a two-stage process. In the first stage, we train an autoencoder to reconstruct the corresponding HDR panorama from an LDR panorama. Following [64], the encoder transforms the LDR skydome panorama into three intermediate vectors, including the peak direction vector $\mathbf{f}_{\text{dir}} \in \mathbb{R}^3$, the intensity vector $\mathbf{f}_{\text{int}} \in \mathbb{R}^3_+$, and the sky content vector $\mathbf{f}_{\text{content}} \in \mathbb{R}^{64}$. However, as HDR intensity behaves like an impulse response at its peak position, with pixel values thousands of times higher than its neighbors, it is difficult for the decoder to recover such patterns. To tackle this, we design a residual connection that injects $\mathbf{f}_{\text{int}}$ into the decoded HDR panorama with a spherical Gaussian lobe attenuation. This explicitly restores the peak intensity of the sun in the reconstructed HDR panorama, allowing us to render strong shadows for virtual objects.

In the second stage, we train an image encoder and a multi-camera fusion module built upon the pretrained decoder from the first stage. Specifically, for images from each camera, a shared image encoder predicts the peak direction vector $\mathbf{f}^{(i)}_{\text{dir}}$, the intensity vector $\mathbf{f}^{(i)}_{\text{int}}$, and the sky content vector $\mathbf{f}^{(i)}_{\text{content}}$ for each image $\mathcal{I}^{(i)}$, where i is the camera index. We design the latent vector fusion across the multiple camera views as follows: all $\mathbf{f}^{(i)}_{\text{dir}}$ are aligned to the front-facing view using their extrinsic parameters and averaged to form $\bar{\mathbf{f}}_{\text{dir}}$; all $\mathbf{f}^{(i)}_{\text{int}}$ are averaged to yield $\bar{\mathbf{f}}_{\text{int}}$; all $\mathbf{f}^{(i)}_{\text{content}}$ are integrated into $\bar{\mathbf{f}}_{\text{content}}$ through a self-attention module. Finally, the pretrained decoder reconstructs the HDR skydome image $\mathcal{I}_{\text{skydome}}$ from $\bar{\mathbf{f}}_{\text{dir}}$, $\bar{\mathbf{f}}_{\text{int}}$ and $\bar{\mathbf{f}}_{\text{content}}$.

Compared to alternative approaches [31, 64], our multi-camera sky dome estimation technique accurately reproduces the sun's intensity response behavior at its peak with our residual designs, significantly improving the accuracy and fidelity of the skydome reconstruction.

Surrounding lighting estimation. Merely modeling the skydome cannot replicate the complex location-specific lighting effects, like those in the shade with sunlight blocked by trees or buildings. Our McNeRF is capable of storing precise 3D scene information, enabling us to capture the surrounding scene's impact on lighting. This approach facilitates the achievement of spatially-varying lighting estimation. Specifically, we sample the hemisphere rays at the virtual object's position $\mathbf{o}$. The rays' directions, $\mathbf{d}_i, i = 0, 1, \cdots, h \times w$, are aligned with pixel coordinates on a unit sphere using equirectangular projection from an environment map, where h and w are map's height and width. With the ray



Command: *"Remove all cars in the scene and add a Porsche driving the wrong way toward me fast. Additionally, add a police car also driving the wrong way and chasing behind the Porsche. The view should be moved 5 meters ahead and 0.5 meters above."*

Figure 5. Editing result under a complex and mixed command.

$\mathbf{r} = \mathbf{o} + t\mathbf{d}_i$, we query our McNeRF as Equation 1 to obtain HDR surrounding lighting $\mathcal{I}_{\text{surround}}(\mathbf{o}, \mathbf{d}_i)$. The surrounding lighting estimation reconstructs complex environmental lighting, achieving a spatially varying effect and high consistency with the background.

Blending. We blend the HDR intensity value from the skydome and surrounding lighting by transmittance of the final sampling point from McNeRF. The idea is that the rays emitted outside the radiance fields will definitely hit the skydome. Given the direction $\mathbf{d}_i$, we retrieve the skydome's intensity $\mathcal{I}_{\text{skydome}}(\mathbf{d}_i)$ with equirectangular projection. The final HDR light intensity $\mathcal{I}_{\text{env}}(\mathbf{o}, \mathbf{d}_i)$ is a combination of scene and skydome:

$$\mathcal{I}_{\text{env}}(\mathbf{o}, \mathbf{d}_i) = \mathcal{I}_{\text{surround}}(\mathbf{o}, \mathbf{d}_i) + T_K \mathcal{I}_{\text{skydome}}(\mathbf{d}_i),$$

where T_K is the last sampling point's transmittance.

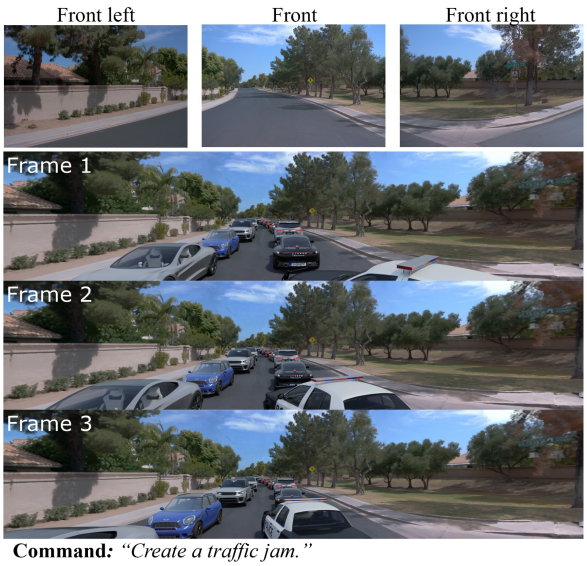
McLight offers two main advantages: i) it explicitly recovers the illuminance behavior at the peak and use complementary information from multiple cameras to restore accurate skydome; and ii) it enables location-specific lighting with consideration of complex scene structures.

5. Experimental Results

5.1. Datasets and implementation details

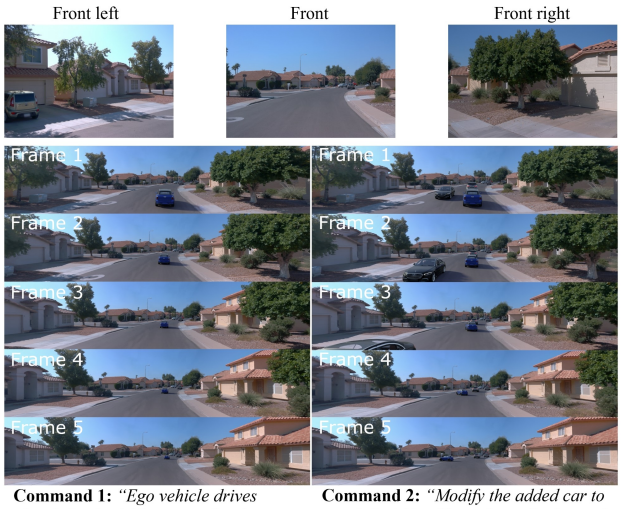
We demonstrate a variety of results mainly on the Waymo Open Dataset [57], which contains high-quality multi-camera images and the corresponding calibrations. For McLight skydome estimation, we collect 449 HDRIs from online HDRI databases for the autoencoder training and use HoliCity [75], a street view panorama dataset for the second stage; see more dataset details in the Appendix.

In our experiment, we use front, left front, and right front



Command: *"Create a traffic jam."*

Figure 6. Editing result under a highly abstract command.



Command 1: *"Ego vehicle drives ahead slowly. Add a car to the close front that is moving ahead."*

Command 2: *"Modify the added car to turn left. Add a Chevrolet to the front of the added car. Add another vehicle to the left of the added Mini driving toward me."*

Figure 7. Editing result under multi-round commands.

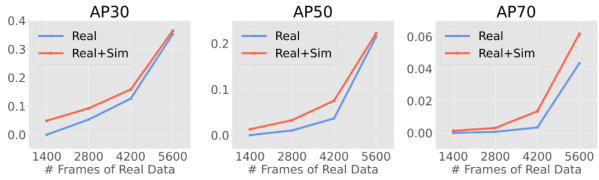


Figure 8. Comparison of detection performance w/o and with our simulated data under different amounts of real data during training.

cameras in each frame. During the rendering process, we choose 40 frames per scene at a 10Hz sampling rate, totaling 120 images. We evenly select $1/8$ of these as the test set, with the remainder used for training. The input images are used at the dataset's initial resolution of 1920×1280; we employ GPT-4 as the LLMs in all of our experiments; see more implementation details in the Appendix.

5.2. System results

Editing via language commands. We select three representative commands to demonstrate the editing results. All of the results demonstrate we achieve photo-realistic wide angle results, thanks to McNeRF and McLight.

Mixed and complex command. We send the system with a mixed and complex command, implying that a police car is chasing a wrong-way racer. The target scene, command and the result are shown in Figure 5. We see that i) every requirement in the complex command is accurately executed thanks to our multi-agent collaboration design; ii) this command successfully simulates one rare but dangerous driving condition, which is significant in accident testing.

Highly abstract command. The second type is a highly abstract command. The inputs and results are presented in Figure 6. We see that i) this highly abstract command is hard to decompose by sentence division but still can be correctly executed by our method, and ii) our 3D asset bank offers a large variety of objects for addition.

Multi-round command. We also perform a multi-round chat with our system, and the commands in different rounds exist context dependencies. The final results are shown in Figure 7. We see that i) our system is well-equipped to handle multi-round commands and execute the commands in each round precisely; ii) our system can handle the context dependencies across different rounds thanks to the recording ability of the project manager agent.

3D detection with simulation data. We validate the benefits of our simulation as data augmentation for a downstream 3D object detection task on Waymo Open Dataset [58]. We simulate 1960 frames, derived from scenes in the training dataset. In the simulation, cars with various types, locations, and orientations are incorporated. The detection model adopts Lift-Splat [50]. Figure 8 shows detection performances with and without fixed augmentation under various amounts of real data. We see that i) a significant and consistent improvement across different data sizes is achieved; ii) when real data is limited, our simulation notably aids in rough detection (AP30); iii) when the amount of real data increases, our simulation further significantly improves fine-grained detection (AP70), reflecting the high-quality of our simulation.

5.3. Component results

Multi-agent collaboration. We evaluated the effectiveness of the multi-agent collaboration by checking whether the command is successfully executed in Table 2. In scenarios without multi-agent collaboration, all operations are executed by a single LLM agent. We see that a single LLM agent leads to notably lower execution accuracy across all categories due to process limitations. In contrast, the collaborative multi-agent approach can manage most commands, attributed to its task division and agent role specificity.

Multi-agent collaboration	Language command category				
	Deletion	Addition	View change	Revision	Abstract
	0.617	0.383	0.717	0.367	0.216
✓	**0.983**	**0.867**	**0.967**	**0.917**	**0.883**

Table 2. The accuracy (%) of task completion by LLM without and with multi-agent collaboration.

Methods	PSNR↑	SSIM↑	LPIPS↓	Inf. time (s)↓
DVGO [56]	23.57	0.770	0.508	7.7
Mip-NeRF360 [8]	24.40	0.754	0.528	101.8
S-NeRF [69]	24.71	0.759	0.519	114.5
F2NeRF [62]	23.26	0.773	0.439	2.4
Ours w/o alignment	23.32	0.776	0.437	2.5
Ours w/o exposure	25.18	0.819	0.381	2.4
McNeRF (Ours)	**25.82**	**0.822**	**0.378**	2.5

Table 3. Background novel view rendering performance evaluation.

Figure 9. Comparisons of wide-angle images generation. (a) S-NeRF.(b) F2NeRF. (c) McNeRF (Ours). Last row: target images.

Background rendering. We compare our McNeRF with several other state-of-the-art methods on the background novel view synthesis task. We perform reconstruction and rendering on 32 selected scenes. Table 3 shows the quantitative results comparison on three metrics: PSNR, SSIM, and LPIPS. We see that i) McNeRF achieves SoTA performance on all three metrics, significantly outperforming other baselines; ii) McNeRF has a fast inference speed, enabling quick responses to user requests for image rendering.

Figure 9 demonstrates qualitative comparisons between other methods and ours. We see that existing NeRF methods do not consider the exposure time, leading to noticeable changes in brightness at the junctions of different cameras in the image, as well as an overall inconsistency in exposure across the wide-angle view. Our method can make the brightness of the entire image more consistent.

Foreground rendering. We compare our McLight with the other two SoTA methods [31, 64]. Table 4 shows the comparison of relative intensity(log 10) error on our HDRI dataset, angular error on HoliCity [75], and user study. We see that

Method	Peak Intensity(log10) Error		Peak Angular Error (deg)		User study(%) ↑
	Mean ↓	Median ↓	Mean ↓	Median ↓	
Hold-Geoffroy et al. [31]	0.899	0.975	48.4	51.6	19.5
Wang et al. [64]	0.590	0.628	33.5	29.4	37.3
McLight (Ours)	**0.449**	**0.270**	**32.3**	**26.5**	**43.1**

Table 4. Comparison with previous methods on lighting estimation.

Figure 10. Comparison with different lighting estimation methods.

Methods	Straight	Left Turn	Right Turn	Speed	Within-road
GPT2Code	0.738	0.559	0.536	0.893	0.214
GPT2Motion	0.595	0.119	0.167	0.345	0.277
Ours	**0.988**	**0.940**	**0.976**	**0.952**	**1.000**

Table 5. Comparison with motion generation from text methods.

McLight achieves more accurate peak behavior and receives noticeably higher user preferences. Figure 10 shows the visualizations of vehicle insertion. The vehicles added through McLight feature significantly more realistic reflections and strong shadows consistent with the scene.

Vehicle motion. As shown in Table 5, we compare the motion generation method from user commands with two of our designed baselines: 1. GPT2Motion, which directly uses LLM to return the motion coordinates; 2. GPT2Code, which first generates code using LLM and executes it to obtain the vehicle motion. We validate multiple actions in multiple scenarios and report the user study result. The user study is to determine if the generated motions matched the command intents and fitted with the lane map. We see that our method demonstrated a significant advantage in generating motions from language commands. Additionally, it maintained a high rate of keeping the trajectories within the lane boundaries.

6. Conclusions and Limitations

This paper introduces *ChatSim*, the first system for editing 3D driving scene simulations via language commands and realistic rendering with import of external digital assets. To effectively execute user commands, *ChatSim* adopts an LLM-agent collaboration workflow. To promote photo-realistic simulation, we propose McNeRF and McLight for background and foreground rendering, respectively, accommodating multi-camera inputs. Experiments show that *ChatSim* successfully simulates customized data via language commands, achieving high-quality, photo-realistic outcomes. **In future**, we plan to integrate more background editing functionalities to *ChatSim*, such as weather changes.

Acknowledgement

This research is supported by NSFC under Grant 62171276 and the Science and Technology Commission of Shanghai Municipal under Grant 21511100900 and 22DZ2229005.

References

[1] 51Sim-One. https://wdp.51aes.com/news/27. 2, 3

[2] LLC Agisoft. Metashape–photogrammetric processing of digital images and 3d spatial data generation, 2019. 5

[3] Elif Akata, Lion Schulz, Julian Coda-Forno, Seong Joon Oh, Matthias Bethge, and Eric Schulz. Playing repeated games with large language models. *arXiv preprint arXiv:2305.16867*, 2023. 3

[4] Alexander Amini, Igor Gilitschenski, Jacob Phillips, Julia Moseyko, Rohan Banerjee, Sertac Karaman, and Daniela Rus. Learning robust control policies for end-to-end autonomous driving from data-driven simulation. *IEEE Robotics and Automation Letters*, 5(2):1143–1150, 2020. 1

[5] Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023. 3

[6] Natalie M Banta. Property interests in digital assets: The rise of digital feudalism. *Cardozo L. Rev.*, 38:1099, 2016. 1

[7] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021. 2

[8] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022. 2, 8

[9] Luca Bergamini, Yawei Ye, Oliver Scheel, Long Chen, Chih Hu, Luca Del Pero, Błażej Osiński, Hugo Grimmett, and Peter Ondruska. Simnet: Learning reactive self-driving simulations from real-world observations. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5119–5125. IEEE, 2021. 1

[10] Mark Boss, Varun Jampani, Kihwan Kim, Hendrik Lensch, and Jan Kautz. Two-shot spatially-varying brdf and shape estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3982–3991, 2020. 3

[11] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 3

[12] 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. 1

[13] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8748–8757, 2019. 1

[14] ChatGPT. https://openai.com/blog/chatgpt. 3

[15] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE international conference on computer vision*, pages 2722–2730, 2015. 1

[16] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2147–2156, 2016.

[17] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017. 1

[18] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022. 3

[19] International Electrotechnical Commission et al. Iec 61966-2-1: 1999 multimedia systems and equipment- colour measurement and management- part 2-1: Colour management-default rgb colour space- srgb, 1999. 5

[20] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 5

[21] Valter Crescenzi, Giansalvatore Mecca, Paolo Merialdo, et al. Roadrunner: Towards automatic data extraction from large web sites. In *VLDB*, pages 109–118, 2001. 2, 3

[22] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017. 2, 3

[23] Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*, 2023. 3

[24] Openscanerio Editor. https://github.com/ebadi/openscenarioeditor. 2, 3

[25] Unreal Engine. https://www.unrealengine.com/. 2

[26] Ruiyuan Gao, Kai Chen, Enze Xie, Lanqing Hong, Zhenguo Li, Dit-Yan Yeung, and Qiang Xu. Magicdrive: Street view generation with diverse 3d geometry control. *arXiv preprint arXiv:2310.02601*, 2023. 2, 3

[27] Mathieu Garon, Kalyan Sunkavalli, Sunil Hadap, Nathan Carr, and Jean-François Lalonde. Fast spatially-varying indoor lighting estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6908–6917, 2019. 3

[28] Jianfei Guo, Nianchen Deng, Xinyang Li, Yeqi Bai, Botian Shi, Chiyu Wang, Chenjing Ding, Dongliang Wang, and Yikang Li. Streetsurf: Extending multi-view implicit surface reconstruction to street views. *arXiv preprint arXiv:2306.04988*, 2023. 2

[29] Rui Hao, Linmei Hu, Weijian Qi, Qingliu Wu, Yirui Zhang, and Liqiang Nie. Chatllm network: More brains, more intelligence. *arXiv preprint arXiv:2304.12998*, 2023. 3

[30] Yannick Hold-Geoffroy, Kalyan Sunkavalli, Sunil Hadap, Emiliano Gambaretto, and Jean-François Lalonde. Deep outdoor illumination estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7312–7321, 2017. 3

[31] Yannick Hold-Geoffroy, Akshaya Athawale, and Jean-François Lalonde. Deep sky modeling for single image outdoor lighting estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6927–6935, 2019. 3, 6, 8

[32] Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, et al. Metagpt: Meta programming for multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*, 2023. 3

[33] Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. Gaia-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023. 2, 3

[34] Abhijit Kundu, Kyle Genova, Xiaoqi Yin, Alireza Fathi, Caroline Pantofaru, Leonidas J Guibas, Andrea Tagliasacchi, Frank Dellaert, and Thomas Funkhouser. Panoptic neural fields: A semantic object-aware neural scene representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12871–12881, 2022. 2

[35] Jean-François Lalonde and Iain Matthews. Lighting estimation in outdoor image collections. In *2014 2nd international conference on 3D vision*, pages 131–138. IEEE, 2014. 3

[36] Jean-François Lalonde, Srinivasa G Narasimhan, and Alexei A Efros. What do the sun and the sky tell us about the camera? *International Journal of Computer Vision*, 88: 24–51, 2010. 3

[37] Chloe LeGendre, Wan-Chun Ma, Graham Fyffe, John Flynn, Laurent Charbonnel, Jay Busch, and Paul Debevec. Deeplight: Learning illumination for unconstrained mobile mixed reality. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5918–5928, 2019. 3

[38] Peixuan Li, Huaici Zhao, Pengfei Liu, and Feidao Cao. Rtm3d: Real-time monocular 3d detection from object keypoints for autonomous driving. In *European Conference on Computer Vision*, pages 644–660. Springer, 2020. 1

[39] Xiaofan Li, Yifu Zhang, and Xiaoqing Ye. Drivingdiffusion: Layout-guided multi-view driving scene video generation with latent diffusion model. *arXiv preprint arXiv:2310.07771*, 2023. 2, 3

[40] Zhengqin Li, Zexiang Xu, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker. Learning to reconstruct shape and spatially-varying reflectance from a single image. *ACM Transactions on Graphics (TOG)*, 37(6):1–11, 2018. 3

[41] Zhuopeng Li, Lu Li, and Jianke Zhu. Read: Large-scale neural scene rendering for autonomous driving. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1522–1529, 2023. 2, 3

[42] Zhijian Liu, Haotian Tang, Alexander Amini, Xinyu Yang, Huizi Mao, Daniela L Rus, and Song Han. Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2774–2781. IEEE, 2023. 4

[43] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 2

[44] Milan Miric, Kevin J Boudreau, and Lars Bo Jeppesen. Protecting their digital assets: The use of formal & informal appropriability strategies by app developers. *Research Policy*, 48(8):103738, 2019. 1

[45] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15, 2022. 2

[46] OpenAI. Gpt-4 technical report. 2023. 3

[47] Julian Ost, Fahim Mannan, Nils Thuerey, Julian Knodt, and Felix Heide. Neural scene graphs for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2856–2865, 2021. 2, 3

[48] Julian Ost, Issam Laradji, Alejandro Newell, Yuval Bahat, and Felix Heide. Neural point light fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18419–18429, 2022. 2, 3

[49] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022. 3

[50] Jonah Philion and Sanja Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, pages 194–210. Springer, 2020. 7

[51] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021. 4

[52] Guodong Rong, Byung Hyun Shin, Hadi Tabatabaee, Qiang Lu, Steve Lemke, Mārtiņš Možeiko, Eric Boise, Geehoon Uhm, Mark Gerow, Shalin Mehta, et al. Lgsvl simulator: A high fidelity simulator for autonomous driving. In *2020 IEEE 23rd International conference on intelligent transportation systems (ITSC)*, pages 1–6. IEEE, 2020. 1

[53] Soumyadip Sengupta, Jinwei Gu, Kihwan Kim, Guilin Liu, David W Jacobs, and Jan Kautz. Neural inverse rendering of an indoor scene from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8598–8607, 2019. 3

[54] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics: Results of the 11th International Conference*, pages 621–635. Springer, 2018. 2, 3

[55] Gowri Somanath and Daniel Kurz. Hdr environment map estimation for real-time augmented reality. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11298–11306, 2021. 3

[56] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022. 2, 8

[57] 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. 6

[58] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454, 2020. 1, 7

[59] Alexander Swerdlow, Runsheng Xu, and Bolei Zhou. Street-view image generation from a bird's-eye view layout. *arXiv preprint arXiv:2301.04634*, 2023. 2, 3

[60] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 3

[61] Haithem Turki, Jason Y Zhang, Francesco Ferroni, and Deva Ramanan. Suds: Scalable urban dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12375–12385, 2023. 2

[62] Peng Wang, Yuan Liu, Zhaoxi Chen, Lingjie Liu, Ziwei Liu, Taku Komura, Christian Theobalt, and Wenping Wang. F2-nerf: Fast neural radiance field training with free camera trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4150–4159, 2023. 2, 5, 8

[63] Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, and Jiwen Lu. Drivedreamer: Towards real-world-driven world models for autonomous driving. *arXiv preprint arXiv:2309.09777*, 2023. 2, 3

[64] Zian Wang, Wenzheng Chen, David Acuna, Jan Kautz, and Sanja Fidler. Neural light field estimation for street scenes with differentiable virtual object insertion. In *European Conference on Computer Vision*, pages 380–397. Springer, 2022. 3, 6, 8

[65] Zhenhailong Wang, Shaoguang Mao, Wenshan Wu, Tao Ge, Furu Wei, and Heng Ji. Unleashing cognitive synergy in large language models: A task-solving agent through multi-persona self-collaboration. *arXiv preprint arXiv:2307.05300*, 2023. 3

[66] Bichen Wu, Forrest Iandola, Peter H Jin, and Kurt Keutzer. Squeezedet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 129–137, 2017. 1

[67] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155*, 2023. 3

[68] Zirui Wu, Tianyu Liu, Liyi Luo, Zhide Zhong, Jianteng Chen, Hongmin Xiao, Chao Hou, Haozhe Lou, Yuantao Chen, Runyi Yang, Yuxin Huang, Xiaoyu Ye, Zike Yan, Yongliang Shi, Yiyi Liao, and Hao Zhao. Mars: An instance-aware, modular and realistic simulator for autonomous driving. *CICAI*, 2023. 2, 3

[69] Ziyang Xie, Junge Zhang, Wenye Li, Feihu Zhang, and Li Zhang. S-nerf: Neural radiance fields for street views. *arXiv preprint arXiv:2303.00749*, 2023. 2, 3, 8

[70] Yinda Xu and Lidong Yu. Drl-based trajectory tracking for motion-related modules in autonomous driving. *arXiv preprint arXiv:2308.15991*, 2023. 4

[71] Kairui Yang, Enhui Ma, Jibin Peng, Qing Guo, Di Lin, and Kaicheng Yu. Bevcontrol: Accurately controlling street-view elements with multi-perspective consistency via bev sketch layout. *arXiv preprint arXiv:2308.01661*, 2023. 2, 3

[72] Zhenpei Yang, Yuning Chai, Dragomir Anguelov, Yin Zhou, Pei Sun, Dumitru Erhan, Sean Rafferty, and Henrik Kretzschmar. Surfelgan: Synthesizing realistic sensor data for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11118–11127, 2020. 1

[73] Ze Yang, Yun Chen, Jingkang Wang, Sivabalan Manivasagam, Wei-Chiu Ma, Anqi Joyce Yang, and Raquel Urtasun. Unisim: A neural closed-loop sensor simulator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1389–1399, 2023. 2, 3

[74] Jinsong Zhang, Kalyan Sunkavalli, Yannick Hold-Geoffroy, Sunil Hadap, Jonathan Eisenman, and Jean-François Lalonde. All-weather deep outdoor lighting estimation. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 10158–10166, 2019. 3

[75] Yichao Zhou, Jingwei Huang, Xili Dai, Linjie Luo, Zhili Chen, and Yi Ma. HoliCity: A city-scale data platform for learning holistic 3D structures. 2020. arXiv:2008.03286 [cs.CV]. 6, 8

[76] Mingchen Zhuge, Haozhe Liu, Francesco Faccio, Dylan R Ashley, Róbert Csordás, Anand Gopalakrishnan, Abdullah Hamdi, Hasan Abed Al Kader Hammoud, Vincent Herrmann, Kazuki Irie, et al. Mindstorms in natural language-based societies of mind. *arXiv preprint arXiv:2305.17066*, 2023. 3