This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

SceneDiffuser++: City-Scale Traffic Simulation via a Generative World Model

Shuhan Tan[2*] John Lambert[1] Hong Jeon[1] Sakshum Kulshrestha[1] Yijing Bai[1]
Jing Luo[1] Dragomir Anguelov[1] Mingxing Tan[1] Chiyu Max Jiang[1]
[1]Waymo LLC [2]UT Austin

Simulation Route (a) Initial Scene Generation (b) Long Simulation Rollout (c) Final Destination

Figure 1. Overview. SceneDiffuser++ is a single unified, end-to-end trained generative world model that enables CitySim: city-scale traffic simulation that takes in a large map region, start and end points, and simulates everything in between, from initial scene generation, agent behavior prediction, occlusion reasoning, dynamic agent generation (spawning and removal) to environment simulation (traffic lights).

Abstract

The goal of traffic simulation is to augment a potentially limited amount of manually-driven miles that is available for testing and validation, with a much larger amount of simulated synthetic miles. The culmination of this vision would be a generative simulated city, where given a map of the city and an autonomous vehicle (AV) software stack, the simulator can seamlessly simulate the trip from point A to point B by populating the city around the AV and controlling all aspects of the scene, from animating the dynamic agents (e.g., vehicles, pedestrians) to controlling the traffic light states. We refer to this vision as CitySim*, which requires an agglomeration of simulation technologies: scene generation to populate the initial scene, agent behavior modeling to animate the scene, occlusion reasoning, dynamic scene generation to seamlessly spawn and remove agents, and environment simulation for factors such as traffic lights. While some key technologies have been separately studied in various works, others such as dynamic scene generation and environment simulation have received less attention in the research community. We propose SceneDiffuser++, the first end-to-end generative world model trained on a single loss function capable of point A-to-B simulation on a city scale integrating all the requirements above. We demonstrate the city-scale traffic simulation capability of SceneDiffuser++ and study its superior realism under long simulation conditions. We evaluate the simulation quality on an augmented version of the Waymo Open Motion Dataset (WOMD) with larger map regions to support trip-level simulation.*

1. Introduction

Imagine an ideal traffic simulation at the city-scale: Starting from a logged or synthetic scene, we initiate the simulation. The virtual world comes alive with agents behaving realistically: cars navigate roads, pedestrians cross streets, and interactions unfold naturally. A pedestrian emerges from behind a bus, prompting a reaction from the ego agent. Vehicles disappear and reappear as they become occluded and disoccluded. Turning onto a new road reveals a fresh stream of traffic. The ego vehicle responds to traffic signals, stopping at red lights and proceeding when they turn green. This simulation persists for a long duration, allowing trip-level evaluations of driving by generating a dynamically populated virtual city with continuous agent interactions.

We refer to such a city-scale closed-loop traffic simulation system as CitySim (See Fig. 1). CitySim can enable point-to-point driving simulation for obtaining trip-level statistics. This allows holistic driving assessment, for instance trip-level travel time comparisons to average hu-

*Work done as an intern at Waymo.

man drivers, pick-up and drop-off quality assessment, as well as evaluation of driving behaviors during the trip, including safety and driving quality. Such simulators also allow for playing out events that take longer to unfold, such as interactions between an AV and emergency vehicles. They can also facilitate pre-release evaluation of AV software by estimating safety and quality related rates, system-level hill-climbing, and system-level fault discovery[1, 10]. CitySim systems stand in contrast to simulation frameworks based on simulating logged events (usually $<$ 10s), which is the mainstream setup in most existing frameworks [14, 32].

Transitioning from event-level simulation to trip-level simulation requires a step-function improvement in simulation capabilities. While event-level simulations are short in duration, naively extending them to longer durations triggers a host of realism issues. In longer simulation, the initial logged agents [4, 5, 12, 51] might leave the periphery of the AV while new agents might continuously and seamlessly appear, mandating dynamic agent generation to handle agent spawning and removal. The need for dynamic agent generation is more critical in cases where the AV takes a different route or speed profile which might result in the AV very quickly turning into an empty street. Furthermore, as the simulated AV heads into regions of the map not traversed in the initial log, traffic light states and other environment factors need to be simulated as well. Simulation artifacts arising from high pose divergence between the logged and simulated AV are referred to as "simulation drift" [2].

These unrealistic behaviors in high pose divergence scenarios highlight three critical, yet often overlooked, capabilities in learned traffic simulation: dynamic agent generation (including *agent spawning* for new agents entering the scene, *agent removal* for agents exiting the scene), *occlusion reasoning*, and the dynamic handling of *critical environmental factors* like traffic lights. To our knowledge, most of the aforementioned technologies are not investigated in existing learned simulation models.

In this work, we bring together this vision of a realistic and dynamically populated virtual city that enables trip-level simulation in a single end-to-end learned, generative world model that we refer to as SceneDiffuser++. SceneDiffuser++ is a diffusion model that is solely trained on the diffusion denoising objective, yet supports all aforementioned capabilities via simple autoregressive rollout. Following Jiang et al. [24], we model the problem as denoising the scene-tensor, with various key insights. First, we observe that agent spawning, removal and occlusion reasoning can be jointly modeled simply via predicting an additional validity (or equivalently, visibility) channel along with other agent features such as x, y, size, type, etc. Though conceptually simple, this requires diffusion to learn to generate sparse tensors without prespecified sparse structure. We propose a simple yet effective training loss formulation and inference-time diffusion sampler modification to allow stable training and sampling of such models. Finally, we propose a novel architecture change that allows simulating the joint rollouts of various non-homogeneous scene elements (e.g., agents and traffic lights with different feature sizes). We propose novel ways to evaluate the realism of such trip-level simulation, and benchmark and ablate our design choices on a version of the Waymo Open Motion Dataset (WOMD) augmented with enlarged kilometer-scale map regions for long rollouts.

In summary, our contributions are as follows:

- We conceptualize the novel city-scale traffic simulation task: CitySim, which focuses on trip-level simulations.
- In contrast to event-level simulations, we identify novel challenges from trip-level simulations, and propose novel evaluation metrics for evaluating the realism of agent spawning, removal, occlusion and traffic light simulation.
- We propose a unified generative world model: SceneDiffuser++, enabling realistic long simulations while accounting for dynamic agent generation, occlusion reasoning and traffic light simulation via simple autoregressive rollout using a novel method to generate sparse tensors.
- We demonstrate our performance on a map-augmented WOMD dataset [12] and achieve state-of-the-art trip-level simulation realism.

2. Related Work

World Models Recently, *world models*, i.e. simulators of how the physical world evolves over time, have attracted significant interest. These AI systems must first build an internal representation of an environment, and then use it to simulate future events within that environment [40], either in pixel representations [3, 21, 49] or abstract latent future simulations [16–18]. The most challenging setting is an *interactive* world model, where the world is rolled out autoregressively [49, 52], rather than in a single shot [3, 37].

Diffusion-Based Traffic Simulation SceneDiffuser [24] demonstrated that a unified model could be used for both scene initialization and closed-loop rollout, and that amortized diffusion [56] could make diffusion more efficient and realistic. However, SceneDiffuser suffers from 3 key limitations. First, it predicts only agent features, rather than other environment features; second, it assumes known agent validity from logged data, limiting simulation duration to the length of logged data in WOMD; third, both the AV and world agents are jointly produced by a single model, potentially introducing collusion. In our work, we address all these limitations. Other methods also use diffusion for open-loop agent simulation [9, 15, 23, 34, 53, 57], closed-loop agent simulation [6, 22], or for initial condition generation [8, 30, 38, 39, 42].

Agent Insertion and Deletion for Simulation Most works

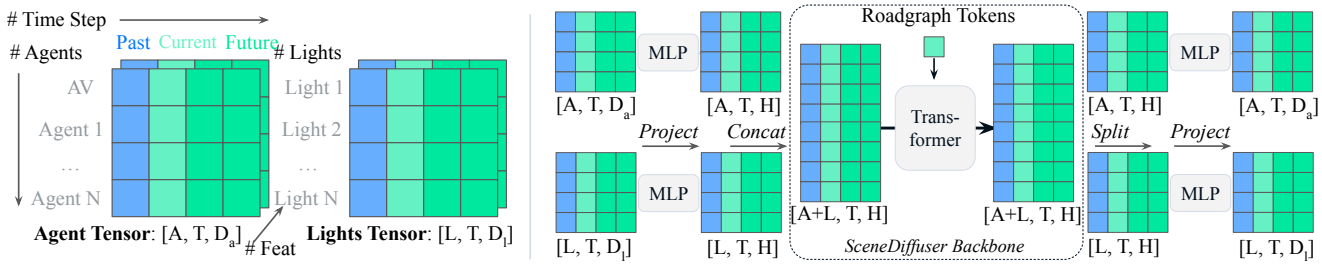


Figure 2. Multi-Tensor Diffusion jointly denoises the set of scene elements in a joint fashion. **Left**: the two scene tensors for agents and traffic lights with a varying number of elements and feature dimensions. **Right**: denoiser architecture for multi-tensors that first projects and homogenizes different scene tensors to the same number of latents before concatenating into a full scene tensor to pass through an axial-attention based transformer backbone [24].

assume a fixed set of agents throughout an entire scene [2, 7, 22, 24, 31, 33–36, 42, 43, 46, 55], reducing simulation to a two-stage process: scene initialization and subsequent rollout. Some of these works insert all agents before the start of the simulation, i.e. for a single fixed timestep, either all at once [38, 42, 45], or sequentially using conditional GANs [2], ConvLSTMs [44], GMMs [13], or predicted occupancy grids [31]. This two-stage factorization approach has two limitations: First, that it causes simulation realism to degrade due to a lack of actor density, and second, that it cannot capture the real-world complexity of evolving and highly dynamic scenes. To our knowledge, no works have studied agent deletion using learned models.

To achieve long-duration simulation, CARLA [11], SUMO [29], and MetaDrive [26] rely on heuristics to insert and delete agents into the scene. For example, MetaDrive [26] procedurally generates maps and spawn points, assigns traffic vehicles to random spawn points on the map, and then recycles them if they stray too far from the AV. However, again, these simplistic heuristics cannot fully capture the complexity and diversity of real world traffic scenes.

Environment Simulation Most methods for autonomous driving simulate only agent behavior and attributes, and not the surrounding environment. The dynamic environment itself can influence world agent and AV behavior, from traffic signals, to weather and time of day, to road hazards, debris, and construction. While data-driven sensor-simulation aims to generate a rendering of the environment [47, 54, 58], we focus instead on a semantic, mid-level representation of the environment, such as traffic light states. To our knowledge, we are the first data-driven method to jointly simulate traffic light signal states and positions, as previous traffic signal control methods assume known signal positions [19, 27, 50]. CARLA [11] controls traffic signals using heuristics.

3. Method

Scene Tensor We denote the scene tensor as $\boldsymbol{x}_i \in \mathbb{R}^{E_i \times \mathcal{T} \times D_i}$, where E_i is the number of elements in the i-th scene tensor (e.g. agents, or traffic lights / signals) jointly modeled in the scene, $\mathcal{T}$ is the total number of modeled physical timesteps, and D is the dimensionality of all the jointly modeled features. We learn to predict attributes for each element: for agents, these are validity v, positional coordinates x, y, z, heading γ, bounding box size l, h, w, and object type $k \in \{\text{AV, car, pedestrian, cyclist}\}$. For traffic lights, these are validity v, positional coordinates x, y, z and a categorical traffic light state s. All features are normalized to $(-1, 1)$ range while agent types are one-hot encoded. All positional coordinates are normalized by the AV's ego pose. We frame all the tasks considered in SceneDiffuser++ as multi-task inpainting tasks on these scene tensors, conditioned on an inpainting mask $\bar{\boldsymbol{m}}_i \in \mathbb{B}^{E_i \times \mathcal{T} \times D_i}$, the corresponding inpainting context values $\bar{\boldsymbol{x}}_i := \bar{\boldsymbol{m}}_i \odot \boldsymbol{x}_i$, and a set of global contexts $\boldsymbol{c}$ (such as roadgraph).

Multi-Tensor We define a multi-tensor $\mathcal{X} := \{\boldsymbol{x}_i\}, \forall i$ as a collection of scene tensors. See Fig. 2 for an illustration of the multi-tensor structure and scene tensors. Without loss of generality, we learn $\mathcal{X} = \{\boldsymbol{x}_{\text{agent}}, \boldsymbol{x}_{\text{light}}\}$ for the joint distribution of agents and traffic lights. We train a diffusion model to learn the conditional probability $p(\mathcal{X}|\mathcal{C})$ where $\mathcal{C} := \{\bar{\boldsymbol{m}}_i, \bar{\boldsymbol{x}}_i, \boldsymbol{c}_i\}, \forall i$. Note that we thus predict a validity mask $\bar{\boldsymbol{v}}_i \in \mathbb{B}^{E_i, \mathcal{T}}$ for a given element (agent or traffic signal) at a given timestep (to account for there being $< E_i$ agents or lights in the scene or for occlusion).

Diffusion Preliminaries We adopt the notation and setup for diffusion models from [20, 24]. Below we denote all scene tensors as $\boldsymbol{x}$ and multi-tensor diffusion is a drop-in replacement of it with $\mathcal{X}$. Forward diffusion gradually adds Gaussian noise to $\boldsymbol{x}$. The noisy scene tensor at diffusion step t can be expressed as $\mathbf{q}(\boldsymbol{z}_t|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}_t|\alpha_t \boldsymbol{x}, \sigma_t^2 \boldsymbol{I})$, where α_t and σ_t are parameters controlling the magnitude and variances of the noise schedule under a variance-preserving model. Therefore $\boldsymbol{z}_t = \alpha_t \boldsymbol{x} + \sigma_t \boldsymbol{\epsilon}_t$, where $\boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \boldsymbol{I})$. We apply the α-cosine schedule where $\alpha_t = \cos(\pi t/2)$ and $\sigma_t = \sin(\pi t/2)$. At peak noise $t = 1$, the forward diffusion process completely destroys the initial scene tensor $\boldsymbol{x}$ resulting in $\boldsymbol{z}_t = \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \boldsymbol{I})$. Assuming

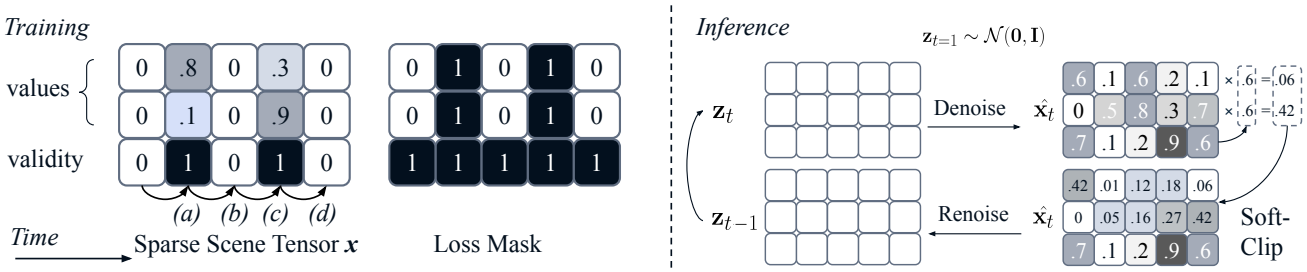


Figure 3. Learning *sparse signals* with diffusion models. We illustrate the scene tensor using one single slice (for a single agent). Learning the validity field in this sparse tensor allows us to model (a) agent spawning, (b) occlusion, (c) disocclusion and (d) removal. **Left**: During training, we impute the corresponding values for all invalid steps to always be zero, before adding noise to train the denoiser. A corresponding loss mask is applied on the diffusion loss. **Right**: During inference, we adopt soft-clipping to multiply the intermediate denoised values by the predicted denoised validity, effectively interpolating with an all zero values vector, weighted by validity confidence.

a Markovian transition process, the transition distribution is $q(\boldsymbol{z}_t|\boldsymbol{z}_s) = \mathcal{N}(\boldsymbol{z}_t|\alpha_{ts}\boldsymbol{z}_s, \sigma_{ts}^2\boldsymbol{I})$, where $\alpha_{ts} = \alpha_t/\alpha_s$ and $\sigma_{ts}^2 = \sigma_t^2 - \alpha_{ts}^2\sigma_s^2$ and $t > s$. The denoising process, conditioned on a single datapoint $\boldsymbol{x}$, can be written as

$$q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}_t|\boldsymbol{\mu}_{t\to s}, \sigma_{t\to s}^2\boldsymbol{I}), \tag{1}$$

where $\boldsymbol{\mu}_{t\to s} = \frac{\alpha_{ts}\sigma_s^2}{\sigma_t^2}\boldsymbol{z}_t + \frac{\alpha_s\sigma_{ts}^2}{\sigma_t^2}\boldsymbol{x}$ and $\sigma_{t\to s} = \frac{\sigma_{ts}^2\sigma_s^2}{\sigma_t^2}$. $\boldsymbol{x}$ is approximated using a learned denoiser $\hat{\boldsymbol{x}}$. Following [20, 24, 41], we adopt the *v prediction* formulation, defined as $\boldsymbol{v}_t(\boldsymbol{\epsilon}_t, \boldsymbol{x}) = \alpha_t\boldsymbol{\epsilon}_t - \sigma_t\boldsymbol{x}$. A model parameterized by $\boldsymbol{\theta}$ is trained to predict $\boldsymbol{v}_t$ from $\boldsymbol{z}_t$, t and context $\mathcal{C}$: $\hat{\boldsymbol{v}}_t := \hat{\boldsymbol{v}}_\theta(\boldsymbol{z}_t, t, \mathcal{C})$. We can recover the predicted $\hat{\boldsymbol{x}}_t$ via $\hat{\boldsymbol{x}}_t = \alpha_t\boldsymbol{z}_t - \sigma_t\hat{\boldsymbol{v}}_t$. The entire model is end-to-end trained with a single loss function:

$$\mathbb{E}_{\substack{(\boldsymbol{x},\mathcal{C})\sim\mathcal{D}, t\sim\mathcal{U}(0,1),\\ \boldsymbol{m}\sim\mathcal{M}, \epsilon_t\sim\mathcal{N}(0,\boldsymbol{I})}}[||(\hat{\boldsymbol{v}}_\theta(\boldsymbol{z}_t, t, \mathcal{C}) - \boldsymbol{v}_t(\boldsymbol{\epsilon}_t, \boldsymbol{x}))\cdot\boldsymbol{w}||_2^2], \tag{2}$$

$\mathcal{D} = \{(\mathcal{X}, \mathcal{C})_j \mid j = 1, 2, \cdots, |\mathcal{D}|\}$ is the dataset containing paired agents and scene context data, t is probabilistically sampled from a uniform distribution, and $\boldsymbol{w} \in \mathbb{B}^{E\times\mathcal{T}\times D}$ is a loss weighting term which we describe in more detail below. $\mathcal{M} = \{\bar{\boldsymbol{m}}_{\text{bp}} \odot \bar{\boldsymbol{m}}_{\text{control}}, \bar{\boldsymbol{m}}_{\text{scenegen}} \odot \bar{\boldsymbol{m}}_{\text{control}}\}$ is the set of inpainting masks for the varied tasks elaborated below.

Tasks We formulate the various tasks, such as scene generation (SceneGen) and behavior prediction (BP) as different inpainting tasks. Following SceneDiffuser [24], the BP inpainting mask $\boldsymbol{m}_{\text{bp}}$ has 1 for all history steps and 0 for all future steps. SceneGen mask $\boldsymbol{m}_{\text{scenegen}}$ consists of 1 for randomly chosen context agents and 0 for agents to predict. The control mask $\boldsymbol{m}_{\text{control}}$ which consists of randomly sampled $\{0, 1\}$, is applied on top of either the SceneGen or BP task for further controllability. This work is a special case of BP with additional agent validity prediction.

Architecture While different event types may differ in the number of entities to predict and their feature dimensions, we adapt the same context encoder and Transformer denoiser backbone architecture as SceneDiffuser [24] by homogenizing different scene tensors. We first project different scene tensors to the same hidden dimension, followed by concatenating along the 'elements' axis (see Fig. 2). After adopting the SceneDiffuser [24] backbone, we apply the reverse process to split and unproject them into the respective scene tensors.

Learning Sparse Tensors One of the major technical contributions of this work is a method for predicting sparse tensors using diffusion. While predicting sparse tensors is of critical importance in this work for learning agent spawning, removal and occlusion, the problem is generic pertaining to learning sparse signals using diffusion models.

Given a sparse tensor $\boldsymbol{x} \in \mathbb{R}^{E\times\mathcal{T}\times D}$, we decompose it into the values: $\mathcal{V}(\boldsymbol{x}) = \boldsymbol{x}[..., : -1] \in \mathbb{R}^{E\times\mathcal{T}\times(D-1)}$ and the validity mask: $\mathcal{M}(\boldsymbol{x}) = \texttt{Clip}(\boldsymbol{x}[..., -1], -1, 1)/2 + 0.5 \in \mathbb{B}^{E\times\mathcal{T}}$. The affine transformation is to reverse the initial normalization. We define its inverse to be $\mathcal{M}^{-1}(\boldsymbol{x}) = 2\boldsymbol{x} - 1$. When the validity is `False`, the corresponding value are arbitrary, whereas when validity is `True`, the corresponding values are meaningful. We seek to jointly predict the values and validity mask. This presents a set of challenges to conventional diffusion model training. How do we supervise the training of values if some corresponding values do not have ground truth (since they are invalid)? Two alternatives arise: (1) impute all invalid values to be zero, and train as if it's a dense tensor, or (2) leave the invalid bits in the values unsupervised. We find that neither of these two approaches can work. Imputation of zeros for invalid values creates significant discontinuities in the signal, leading to unstable model training. Alternatively, if one leaves the invalid bits unsupervised, they are recurrently fed into the denoiser at inference time, leading to very rapid slippage into out-of-distribution values.

We implement a simple yet effective alternative to these approaches, as described in Fig. 3. We first cast all values corresponding to invalid steps to zero with $\boldsymbol{x} \leftarrow \boldsymbol{x} \cdot \mathcal{M}(\boldsymbol{x})$. Then we compute the loss based on Eqn. 2. For weight $\boldsymbol{w}$, we apply the loss mask in as illustrated in Fig. 3, where all features in valid steps are supervised, and only the validity feature in invalid steps are supervised. Dur-

ing inference, we first sample $\boldsymbol{z}_{t=1} \sim \mathcal{N}(0, \boldsymbol{I})$. Then at each denoising step, we first predict the denoised solution at that step: $\hat{\boldsymbol{x}}_t = \alpha_t \boldsymbol{z}_t - \sigma_t \hat{\boldsymbol{v}}_\theta(\boldsymbol{z}_t, t, \mathcal{C})$. Then we apply a step we coin *soft clipping* which we describe below. Finally, we renoise the result to a lower noise level $t-1$ with $\boldsymbol{z}_{t-1} \sim \mathcal{N}(\boldsymbol{z}_{t-1} | \alpha_{t-1}\hat{\boldsymbol{x}}_t, \sigma_{t-1}^2 \boldsymbol{I})$.

We find the *clipping* step to be the most crucial inference-time trick for generating sparse tensors. We contrast soft clipping with several variants:

- soft clipping: $\hat{\boldsymbol{x}}_t \leftarrow \texttt{Concat}(\mathcal{V}(\hat{\boldsymbol{x}}_t) * \mathcal{M}(\hat{\boldsymbol{x}}_t), \mathcal{M}(\hat{\boldsymbol{x}}_t))$
- no clipping: $\hat{\boldsymbol{x}}_t \leftarrow \hat{\boldsymbol{x}}_t$
  Given the two clipping functions for mask and value:
  $\boldsymbol{m}_{clipped} \leftarrow \mathcal{M}^{-1}(\texttt{Where}(\mathcal{M}(\hat{\boldsymbol{x}}_t) < 0.5, \mathbf{0}, \mathbf{1}))$
  $\boldsymbol{v}_{clipped} \leftarrow \texttt{Where}(\mathcal{M}(\hat{\boldsymbol{x}}_t) < 0.5, \mathbf{0}, \mathcal{V}(\hat{\boldsymbol{x}}_t))$
- hard clipping: $\hat{\boldsymbol{x}}_t \leftarrow \texttt{Concat}(\mathcal{V}(\hat{\boldsymbol{x}}_t), \boldsymbol{m}_{clipped})$
- hard-validity clip.: $\hat{\boldsymbol{x}}_t \leftarrow \texttt{Concat}(\boldsymbol{v}_{clipped}, \boldsymbol{m}_{clipped})$

We show in Sec. 4.3 that soft clipping is the most effective strategy that allows stable training and inference with minimal additional changes.

4. Experiment

4.1. Trip-level Traffic Simulation Setup

Dataset We use the Waymo Open Motion Dataset (WOMD)[12] for our trip-level traffic simulation experiments. WOMD includes tracks of all agents and corresponding vectorized maps in each scenario, and it offers a large quantity of high-fidelity object behaviors and shapes produced by a state-of-the-art offboard perception system. Each scenario in WOMD consists of 91 timesteps with a frequency of 10Hz, leading to a 9.1 second scenario. Although the scenario clips are much shorter than our trip-level simulation route, they contain all the critical agent behaviors (driving, entering, exiting, occlusion) and traffic light states (position detection, lane association, state changes). This feature allows us to use these short clips to train SceneDiffuser++ models that can simulate trip-level scenarios much longer than 9.1 seconds. However, during trip-level rollouts ($> 9.1s$), we note that agents easily run out of the map and roadgraph extent, as the original WOMD dataset only contains map regions that cover where the AV can reach in 9.1 seconds. To conduct trip-level simulation, we asked for expanded maps from the WOMD dataset creators (all map elements within circles of 1km radius around any portion of the AV's trajectory) to generate a map-extended dataset that we call *WOMD-XLMap*.

World Model vs. Planner Real simulation use cases require interaction between two disjoint models – a planner and a simulator (world model) [25, 32, 35]. Specifically, the planner controls the AV's movement given the environment and other agents' movement. On the other hand, the world model controls the traffic lights and all other background agents' movement given the AV's movement. At each rollout step, the planner can only observe the world model's history output and cannot obtain its future predictions, and vice versa for world model. In other words, the planner and world model observe each other's predictions only after we rollout their predictions in the environment. In the case where we use the same method as both planner and world model, we ensure they do not share the same predictions by setting different seeds for random sampling.

Method Comparisons We first compare with the *SceneDiffuser*[24] model. Unlike SceneDiffuser++, SceneDiffuser does not model agent validity nor traffic light features. Therefore, during prolonged rollouts with SceneDiffuser, we simply assume that all the agents valid at the current step will remain valid in the future, while setting any future traffic light features as invalid. We also compare with the Intelligent Driver Model (*IDM*) [14, 48] model. To set the routes for IDM to drive for each agent, we start with each agent's initial location and randomly select a valid path with the lane graph on the map. For validity, we set all the agents' future validity to remain the same as their current validity. In our main experiment, we test each possible combination of planner and world model using the three methods.

Metrics For long rollouts, we end up with significant divergence between the logged scene and the propagated scene rollout. Accordingly, it does not make sense to constrain the simulation to adhere closely to the logged data, as done in WOSAC [32]. We also have no 1:1 correspondence between agents, as agents may enter and exit the scene freely.

We use a sliding evaluation window over temporal segments of our long-duration rollouts and ensure that each window has the same temporal length as the log scenario. Then, at each temporal window, we collect the simulated metric value (e.g., number of valid agents) from all simulated scenarios to a list of sim metrics. We also collect all the metric values for the log data to a list of log metrics. We fit two histograms to the sim and log metric values. To measure the realism of the sim features, we compute the Jensen–Shannon (JS) Divergence [28] between these histograms. Lower divergence between histograms indicates more realistic simulated scenarios. Then we compute the mean value over all the divergence values for all windows.

Here we introduce the features over which we compute distributional metrics in our experiments: 1) *# Valid Agents*: the number of agents that have at least one timestep that is valid in the scenario window; 2) *# Entering/Exiting Agents*: the number of agents that are inserted or removed during the scenario window, respectively; 3) *# Entering/Exiting Distance*: the distance to the AV of the entering or exiting agents at the first or last valid timestep in the scenario, respectively; 4) *Offroad Rate*: the fraction of all valid agents located offroad (e.g., in parking lots); 5) *Collision Rate*: the fraction of all valid agents that ever collide with other agents; 6) *Average Speed*: the average speed for all the valid

Table 1. Simulation realism under long rollouts (60s). Numbers are JS-divergence between simulated and logged distributions (↓). Composite is the average of all metrics except TL Violation and TL Transition.

World Model	Planner	# Valid Agents	# Entering Agents	# Exiting Agents	Entering Distance	Exiting Distance	Offroad Rate	Collision Rate	Average Speed	TL Violation	TL Transition	Composite
IDM	IDM	0.4028	0.6357	0.5125	0.3780	0.5253	0.3578	**0.3652**	0.6570	-	-	0.4793
SceneDiffuser	IDM	0.5701	0.7027	0.5767	0.3830	0.3296	0.2765	0.3778	0.6213	-	-	0.4797
SceneDiffuser++	IDM	**0.3132**	**0.1947**	**0.2059**	**0.1620**	**0.1549**	**0.2428**	0.4361	**0.5908**	0.1582	0.0589	**0.2878**
IDM	SceneDiffuser	0.2941	0.7331	0.7279	-	-	**0.0846**	**0.1017**	0.4917	-	-	-
SceneDiffuser	SceneDiffuser	0.4532	0.7114	0.6275	-	0.2759	0.2056	0.3217	0.4036	-	-	-
SceneDiffuser++	SceneDiffuser	**0.2206**	**0.1409**	**0.1526**	**0.1668**	**0.1494**	0.1345	0.4940	**0.3858**	0.1596	0.0264	**0.2306**
IDM	SceneDiffuser++	0.3967	0.6255	0.5170	0.5250	0.5384	0.2056	**0.2990**	**0.2840**	-	-	0.4234
SceneDiffuser	SceneDiffuser++	0.5373	0.6921	0.5718	0.3746	0.2514	0.2384	0.3812	0.4562	-	-	0.4389
SceneDiffuser++	SceneDiffuser++	**0.3053**	**0.2120**	**0.2085**	**0.1183**	**0.1094**	**0.1595**	0.4194	0.3061	0.1625	0.0448	**0.2423**

agents in the scenario window; 7) *TL Violation*: the fraction of all valid agents that violate traffic light rules; 8) *TL Transition*[1]: the transition probability between different traffic light states (e.g., from red to green). Finally, we compute a *Composite* score that is the average of all the metrics.

Simulation Configuration We follow the "Full AR" inference scheme of SceneDiffuser [24]. We vary two key simulation parameters: 1) # rollout steps: the total number of timesteps to rollout, and 2) # replan steps: the number of timesteps between each planner / world model replanning. The smaller the # replan steps, the more frequently the planner and world model are executed and interact with each other. In our main experiments, we set # rollout steps = 600 (60 seconds @ 10Hz) and # replan steps = 40, but we also explore the effects of replan frequencies in Sec. 4.3. Please refer to the Appendix for training and model details.

4.2. Main Results

We show the main result of trip-level traffic simulation in Table 1. We group different experiment settings by which planner is used, and compare the metric results for the rollouts using different world models. Note that some entries are not available in this table. Because SceneDiffuser and IDM do not insert agents into the scene after the first scenario window, their Entering Distance and Exiting Distance results are poor, as expected; accordingly, when using SceneDiffuser Planner, IDM and SceneDiffuser world models, we don't report their entering distance in Table 1.

Our model achieves significantly better performance in all metrics that relate to agent insertion and removal. For example, when we use IDM as the planner, using SceneDiffuser++ as the world model leads to much more realistic distributions of the number of valid, entering and exiting agents, as well as entering and exiting agents' distances. These results indicate that SceneDiffuser++ yields superior performance for predicting when and where to insert and remove agents. In contrast, using IDM or SceneDiffuser models leads to much higher divergence between real and simulated distributions. This result shows it is necessary to predict agent validity for trip-level simulation.

We observe that using our model as a world model leads to, in aggregate, better Average Speed likelihood of the scenario. This is mainly due to the fact that our model is able to predict realistic agent insertion and removal. When agents are able to dynamically appear and exit the scenario, we allow the model to focus more on realistic agent behaviors, e.g. their speed. On the other hand, when the model has to predict features for all the agents that appear in the current step in the future, it has to maintain all the agents' proximity to the AV. Consequently, for SceneDiffuser, we observe all the agents tend to become static during trip-level simulation. We show this effect in Figure 4.

We also note that the Offroad Rate and Collision Rate of our model when used as a world model is worse than that of using IDM and SceneDiffuser. There are a few reasons for this performance: First, during rollout, SceneDiffuser++ will insert agents into the scenario, regardless of how the planner drives before the next replan step. Therefore, it is possible that SceneDiffuser++ will insert agents onto the route of the planner in future steps. However, if no agents are inserted in the scenario (for IDM and SceneDiffuser), all the agents will follow their historic trajectories and drive on safe routes, meaning collisions are less likely. We show in Table 2 that with more frequent replanning, our model leads to a much better collision rate metric. Additionally, we found that SceneDiffuser++ tends to insert a large amount of agents in parking lots that stay parked. These generated parked agents lead to worse offroad metrics.

4.3. Additional Analysis

Inference-time Clipping In Table 3, we present an ablation on soft vs. hard vs. hard-validity clipping for generating sparse tensors. We observe that only soft clipping of features leads to favorable distributions of the number of valid, entering and exiting agents. This indicates that any hard clipping with the validity value on the feature will render the model unable to reflect the agent validity distribution. In addition, we also show the results of a model trained to directly predict invalid agents' features to be 0, and not using clipping during inference (third row), but find this method leads to a higher collision rate, offroad rate, and TL violation rate, along with inferior TL transitions, demonstrating unstable feature prediction values.

Simulation Configurations In Table 2, we compare the results when using SceneDiffuser++ as both world model and

[1]For *TL Transition*, we directly compute the divergence between log and sim transition probability matrices computed over all scenarios.

Table 2. Controlled evaluation of simulation configurations. SceneDiffuser++ serves as both planner and world model.

# Rollout Steps	# Replan Steps	# Valid Agents	# Entering Agents	# Exiting Agents	Entering Distance	Exiting Distance	Offroad Rate	Collision Rate	Average Speed	TL Violation	TL Transition	Composite
600	10	0.3463	0.2308	0.2199	0.0952	0.1231	0.1581	**0.3118**	**0.2681**	**0.1526**	0.0584	**0.1867**
600	20	0.3286	0.2211	0.2165	**0.0910**	**0.1030**	0.1729	0.3702	0.2872	0.1539	**0.0448**	0.1937
600	80	**0.2775**	**0.1853**	**0.1840**	0.1522	0.1356	**0.1461**	0.4478	0.3204	0.1845	0.0668	0.2102
300	40	**0.2526**	**0.1947**	**0.1860**	**0.1195**	**0.1075**	**0.1165**	0.4128	**0.2687**	**0.1579**	0.0396	**0.1936**
1200	40	0.3457	0.2268	0.2259	0.1195	0.1129	0.1950	0.4172	0.3284	0.1715	**0.0312**	0.2213
3000	40	0.4018	0.2758	0.2736	0.1248	0.1235	0.2084	**0.4104**	0.2993	0.2243	0.0468	0.2468

Table 3. Controlled evaluation of SceneDiffuser++ inference time validity decoding strategies, as measured by JS Divergence (↓).

Prediction Mode	# Valid Agents	# Entering Agents	# Exiting Agents	Entering Distance	Exiting Distance	Offroad Rate	Collision Rate	Average Speed	TL Violation	TL Transition	Composite
Hard Clipping	0.4927	0.4776	0.4094	**0.1156**	0.1245	**0.0992**	**0.2602**	**0.2664**	0.2099	**0.0429**	0.2498
Hard-Validity Clipping	0.5963	0.6510	0.5502	0.1741	0.1641	0.2072	0.2830	0.2780	0.2379	0.0435	0.3185
No Clipping	**0.2426**	**0.2035**	0.2139	0.1425	0.1029	0.3026	0.6697	0.3685	0.3123	0.1044	0.2663
Soft Clipping	0.3053	0.2120	**0.2085**	0.1183	**0.1094**	0.1595	0.4194	0.3061	**0.1625**	0.0448	**0.2046**

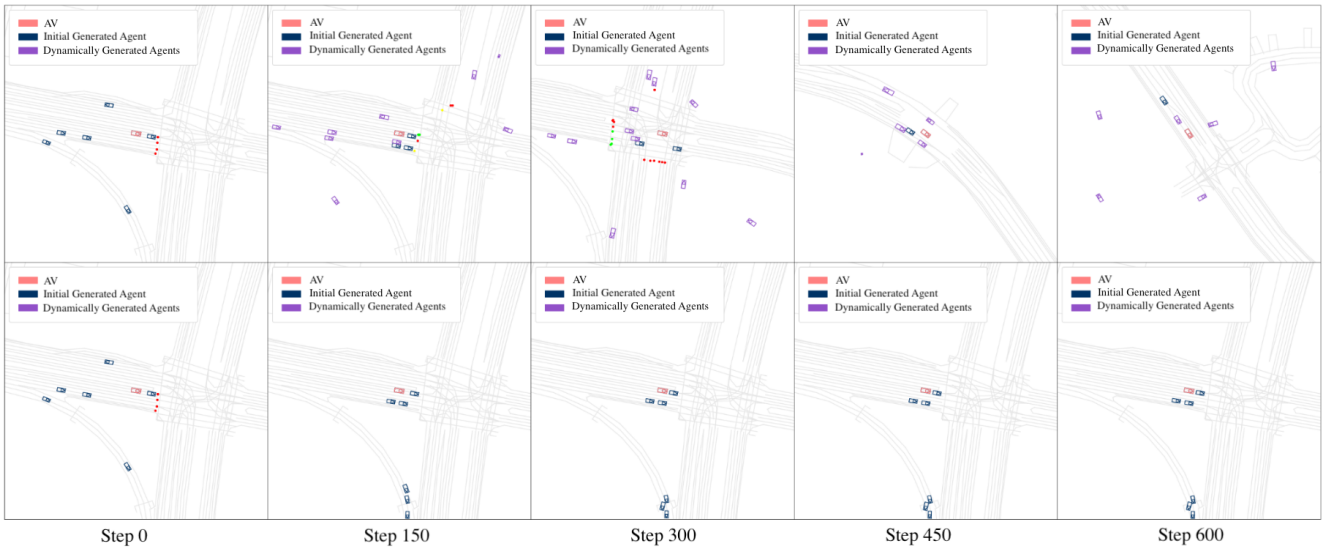


Figure 4. Visualization of a 60s rollout with SceneDiffuser++ (top row) vs. SceneDiffuser, where all agents become stuck in a single location (bottom row). Note that SceneDiffuser++ predicts a variable number of traffic lights to be observable (visible) at any timestep; this is also observed in logged data at times.

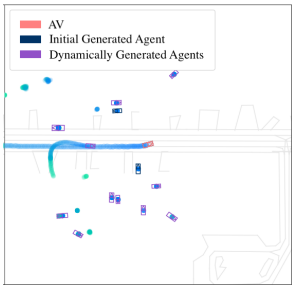
planner under different # rollout steps and # replan steps.

We first show a comparison of different replan steps in the first three rows. We observe that: 1) with more frequent replanning (smaller # replan steps), our model achieves better Collision Rate and Average Speed. This is because when the world model and planner can interact more frequently, they are more reactive to each other's behavior. 2) with less frequent replanning (larger # replan steps), our model leads to better agent insertion and removal behavior. With less frequent replanning, the world model has more timesteps to control into the future, which allows SceneDiffuser++ to better plan when and where to insert agents over the full sequence. On the other hand, with a high replanning rate, only agents that will be predicted to enter into the scenario in the first few timesteps will be inserted, leading to an inferior distribution of agent validity.

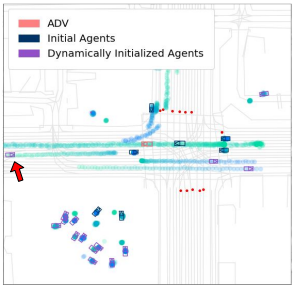
In the next three rows, we show the ablation with the same replan frequency, but different planning horizons, from 30 seconds to 300 seconds. We observe that overall, the realism metrics drop when the model is rolled out over longer horizons. This is mainly due to error from the autoregressive rollout aggregating over time when rolling out over long horizons. Note that although the realism of the number of entering and exiting agents degrades over time, the respective entering or exiting distances stay quite stable. This might indicate that the aggregated error affects agent insertion timing more than insertion position.

Metric Window Curves In Fig. 7, we plot the values of two metrics in Table 1 over simulation timesteps for all world model methods using SceneDiffuser++ as planner. Our model achieves the best performance over all timesteps.

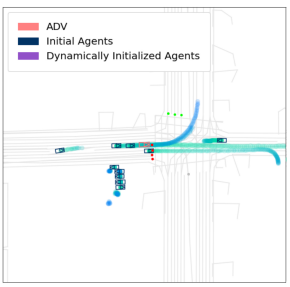
Qualitative Results In Fig. 4 we show examples from 60-second rollouts of our model vs. SceneDiffuser, where we uniformly sample 5 frames from the total 600 steps of each rollouts. For both our model and SceneDiffuser, we roll out using the same model for both world model and planner. It is obvious that our model achieves a realistic trip-level rollout across a large map area with dynamic traffic lights, while SceneDiffuser gets stuck in the starting location, as seen in Figure 4. This is mainly due to two reasons: 1)



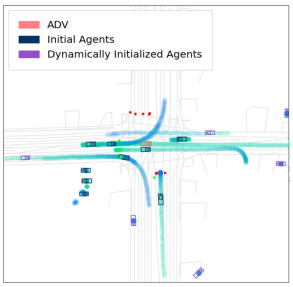
(a) Inserted agent exits a parking lot.



(b) Inserted agent far away from AV.



(c) Inserted agents stop at red light.



(d) Agents continue when light is green.

Figure 5. Plots of SceneDiffuser++ results (green→blue indicates temporal progression).

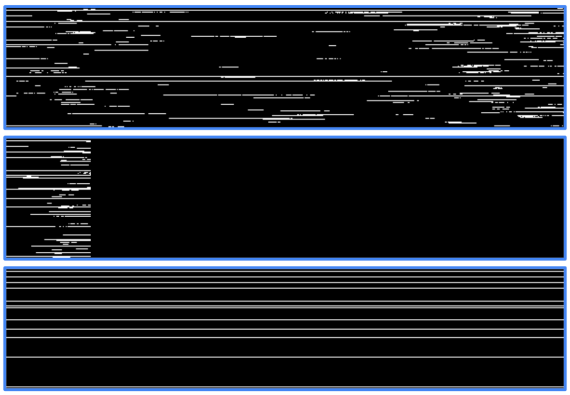


Figure 6. Plot of predicted validity for all 128 agents across 600 steps. X-axis is time, Y-axis is agent ID. White indicates that the agent is valid at that timestep. Top: SceneDiffuser++, Middle: Ground-truth Log (91 steps), Bottom: IDM, SceneDiffuser.

SceneDiffuser does not predict future traffic light location and states, leading to confusion of the AV in the intersection without any traffic lights. 2) SceneDiffuser does not model agents exiting the scenario, therefore it is forced to keep all the agents within the visible range of the AV. Note how the agents in the bottom of the scenario were forced to unrealistically stop in order to keep themselves in the scenario. In comparison, SceneDiffuser++ deals with these issues with a unified model and makes trip-level simulation possible.

Next, we display realistic generated agent behaviors. In Fig. 5(a) we show that agents inserted into a parking lot can realistically navigate onto the main road and merge into traffic. Fig. 5(c) and Fig. 5(d) show that inserted agents comply with traffic light rules, indicating high realism of agent and traffic light interaction. Lastly, Fig. 5(b) shows generated agents can be inserted far from the AV.

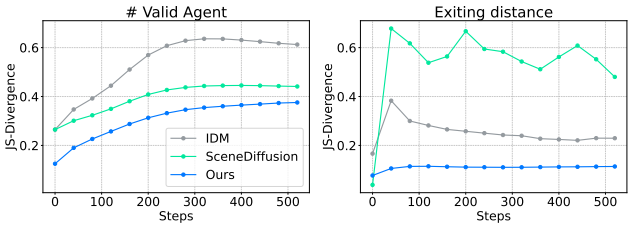


Figure 7. Divergence over simulation steps curve (lower is better).

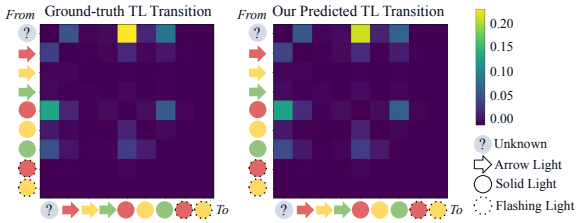


Figure 8. Traffic light transition probability matrix for log vs ours.

Validity Prediction We visualize the predicted agent validity through 600 steps in Fig. 6, comparing our model's output, the logged 'ground truth', and the pattern that IDM and SceneDiffuser both produce. Our model is able to insert and remove agents with realistic validity patterns that are very close to the ground-truth in the first 91 steps. On the other hand, IDM and SceneDiffuser only follow the last-step history validity, leading to a quite unnatural validity pattern. Finally, note that our model is able to insert a new agent to an agent row that was previously occupied by a removed agent (e.g., the last few rows), as long as the previous agent was removed longer ago than SceneDiffuser++'s history horizon. In this way, our method is able to insert any number of agents beyond the total number of agent indices by reusing any agent index where an agent was removed.

Traffic Light Transition We visualize the traffic light state transition probability matrix in Figure 8, where left is the logged ground-truth and right is SceneDiffuser++ prediction. In these figures, along the y-axis is the starting traffic light state and along the x-axis the ending traffic light state. We observe that SceneDiffuser++ traffic light state predictions rigorously follow the ground-truth state transition probability. Note that we remove all the state self-transitions (the diagonal entries) for clearer visualization.

5. Conclusion

We have introduced SceneDiffuser++, a scene-level diffusion prior designed for city-scale traffic simulation. SceneDiffuser++ is a unified world model that enables trip-level long simulations with dynamic agent generation, occlusion reasoning, removal and traffic light simulation. We demonstrate SceneDiffuser++ has strong performance for long-term traffic simulation. We hope our work leads to more realistic trip-level simulation to improve AV safety.

References

[1] Gul Agha and Karl Palmskog. A survey of statistical model checking. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 28(1):1–39, 2018. 2

[2] Luca Bergamini, Yawei Ye, Oliver Scheel, Long Chen, Chih Hu, Luca Del Pero, Błażej Osiński, Hugo Grimmet, and Peter Ondruska. SimNet: Learning reactive self-driving simulations from real-world observations. In *ICRA*, 2021. 2, 3

[3] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024. 2

[4] 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 *CVPR*, pages 11621–11631, 2020. 2

[5] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, and James Hays. Argoverse: 3d tracking and forecasting with rich maps. In *CVPR*, 2019. 2

[6] Wei-Jer Chang, Francesco Pittaluga, Masayoshi Tomizuka, Wei Zhan, and Manmohan Chandraker. Controllable safety-critical closed-loop traffic simulation via guided diffusion, 2023. 2

[7] Wei-Jer Chang, Francesco Pittaluga, Masayoshi Tomizuka, Wei Zhan, and Manmohan Chandraker. Safe-sim: Safety-critical closed-loop traffic simulation with diffusion-controllable adversaries. In *ECCV*, 2024. 3

[8] Kashyap Chitta, Daniel Dauner, and Andreas Geiger. Sledge: Synthesizing driving environments with generative models and rule-based traffic. In *ECCV*, 2024. 2

[9] Younwoo Choi, Ray Coden Mercurius, Soheil Mohamad Alizadeh Shabestary, and Amir Rasouli. Dice: Diverse diffusion model with scoring for trajectory prediction, 2023. 2

[10] Anthony Corso, Robert Moss, Mark Koren, Ritchie Lee, and Mykel Kochenderfer. A survey of algorithms for black-box safety validation of cyber-physical systems. *Journal of Artificial Intelligence Research*, 72:377–428, 2021. 2

[11] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *CoRL*, 2017. 3

[12] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R. Qi, Yin Zhou, Zoey Yang, Aurélien Chouard, Pei Sun, Jiquan Ngiam, Vijay Vasudevan, Alexander McCauley, Jonathon Shlens, and Dragomir Anguelov. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *ICCV*, 2021. 2, 5

[13] Lan Feng, Quanyi Li, Zhenghao Peng, Shuhan Tan, and Bolei Zhou. Trafficgen: Learning to generate diverse and realistic traffic scenarios. In *ICRA*, 2023. 3

[14] Cole Gulino, Justin Fu, Wenjie Luo, George Tucker, Eli Bronstein, Yiren Lu, Jean Harb, Xinlei Pan, Yan Wang, Xiangyu Chen, John D. Co-Reyes, Rishabh Agarwal, Rebecca Roelofs, Yao Lu, Nico Montali, Paul Mougin, Zoey Yang, Brandyn White, Aleksandra Faust, Rowan McAllister, Dragomir Anguelov, and Benjamin Sapp. Waymax: An accelerated, data-driven simulator for large-scale autonomous driving research. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 2023. 2, 5

[15] Zhiming Guo, Xing Gao, Jianlan Zhou, Xinyu Cai, and Botian Shi. SceneDM: Scene-level multi-agent trajectory generation with consistent diffusion models, 2023. 2

[16] David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *NeurIPS*, 2018. 2

[17] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *ICLR*, 2020.

[18] Danijar Hafner, Timothy P Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In *ICLR*, 2021. 2

[19] Pan He, Quanyi Li, Xiaoyong Yuan, and Bolei Zhou. A holistic framework towards vision-based traffic signal control with microscopic simulation. *arXiv preprint arXiv:2403.06884*, 2024. 3

[20] Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. In *ICML*, pages 13213–13232. PMLR, 2023. 3, 4

[21] 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

[22] Zhiyu Huang, Zixu Zhang, Ameya Vaidya, Yuxiao Chen, Chen Lv, and Jaime Fernández Fisac. Versatile scene-consistent traffic scenario generation as optimization with diffusion, 2024. 2, 3

[23] Chiyu "Max" Jiang, Andre Cornman, Cheolho Park, Benjamin Sapp, Yin Zhou, and Dragomir Anguelov. Motion-diffuser: Controllable multi-agent motion prediction using diffusion. In *CVPR*, 2023. 2

[24] Chiyu Max Jiang, Yijing Bai, Andre Cornman, Christopher Davis, Xiukun Huang, Hong Jeon, Sakshum Kulshrestha, John Lambert, Shuangyu Li, Xuanyu Zhou, Carlos Fuertes, Chang Yuan, Mingxing Tan, Yin Zhou, and Dragomir Anguelov. Scenediffuser: Efficient and controllable driving simulation initialization and rollout. In *NeurIPS*, 2024. 2, 3, 4, 5, 6

[25] Napat Karnchanachari, Dimitris Geromichalos, Kok Seang Tan, Nanxiang Li, Christopher Eriksen, Shakiba Yaghoubi, Noushin Mehdipour, Gianmarco Bernasconi, Whye Kit Fong, Yiluan Guo, and Holger Caesar. Towards learning-based planning:the nuplan benchmark for real-world autonomous driving. In *ICRA*, 2024. 5

[26] Quanyi Li, Zhenghao Peng, Lan Feng, Qihang Zhang, Zhenghai Xue, and Bolei Zhou. Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning. *TPAMI*, 2022. 3

[27] Xiaoyuan Liang, Xunsheng Du, Guiling Wang, and Zhu Han. A deep reinforcement learning network for traffic light

cycle control. *IEEE Transactions on Vehicular Technology*, 68(2):1243–1253, 2019. 3

[28] Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*, 37(1):145–151, 1991. 5

[29] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. Microscopic traffic simulation using sumo. In *2018 21st international conference on intelligent transportation systems (ITSC)*, pages 2575–2582. IEEE, 2018. 3

[30] Jack Lu, Kelvin Wong, Chris Zhang, Simon Suo, and Raquel Urtasun. Scenecontrol: Diffusion for controllable traffic scene generation. In *ICRA*, 2024. 2

[31] Reza Mahjourian, Rongbing Mu, Valerii Likhosherstov, Paul Mougin, Xiukun Huang, Joao Messias, and Shimon Whiteson. Unigen: Unified modeling of initial agent states and trajectories for generating autonomous driving scenarios. In *ICRA*, 2024. 3

[32] Nico Montali, John Lambert, Paul Mougin, Alex Kuefler, Nick Rhinehart, Michelle Li, Cole Gulino, Tristan Emrich, Zoey Yang, Shimon Whiteson, Brandyn White, and Dragomir Anguelov. The waymo open sim agents challenge. In *Advances in Neural Information Processing Systems Track on Datasets and Benchmarks*, 2023. 2, 5

[33] Jiquan Ngiam, Vijay Vasudevan, Benjamin Caine, Zhengdong Zhang, Hao-Tien Lewis Chiang, Jeffrey Ling, Rebecca Roelofs, Alex Bewley, Chenxi Liu, Ashish Venugopal, David J Weiss, Benjamin Sapp, Zhifeng Chen, and Jonathon Shlens. Scene transformer: A unified architecture for predicting future trajectories of multiple agents. In *ICLR*, 2022. 3

[34] Matthew Niedoba, Jonathan Lavington, Yunpeng Liu, Vasileios Lioutas, Justice Sefas, Xiaoxuan Liang, Dylan Green, Setareh Dabiri, Berend Zwartsenberg, Adam Scibior, and Frank Wood. A diffusion-model of joint interactive navigation. In *NeurIPS*, 2023. 2

[35] Zhenghao Peng, Wenjie Luo, Yiren Lu, Tianyi Shen, Cole Gulino, Ari Seff, and Justin Fu. Improving agent behaviors with rl fine-tuning for autonomous driving. In *ECCV*, 2024. 5

[36] Jonah Philion, Xue Bin Peng, and Sanja Fidler. Trajeglish: Learning the language of driving scenarios. In *ICLR*, 2024. 3

[37] Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, David Yan, Dhruv Choudhary, Dingkang Wang, Geet Sethi, Guan Pang, Haoyu Ma, Ishan Misra, Ji Hou, Jialiang Wang, Kiran Jagadeesh, Kunpeng Li, Luxin Zhang, Mannat Singh, Mary Williamson, Matt Le, Matthew Yu, Mitesh Kumar Singh, Peizhao Zhang, Peter Vajda, Quentin Duval, Rohit Girdhar, Roshan Sumbaly, Sai Saketh Rambhatla, Sam Tsai, Samaneh Azadi, Samyak Datta, Sanyuan Chen, Sean Bell, Sharadh Ramaswamy, Shelly Sheynin, Siddharth Bhattacharya, Simran Motwani, Tao Xu, Tianhe Li, Tingbo Hou, Wei-Ning Hsu, Xi Yin, Xiaoliang Dai, Yaniv Taigman, Yaqiao Luo, Yen-Cheng Liu, Yi-Chiao Wu, Yue Zhao, Yuval Kirstain, Zecheng He, Zijian He, Albert Pumarola, Ali Thabet, Artsiom Sanakoyeu, Arun Mallya, Baishan Guo, Boris Araya, Breena Kerr, Carleigh Wood, Ce Liu, Cen Peng, Dimitry Vengertsev, Edgar Schonfeld, Elliot Blanchard, Felix Juefei-Xu, Fraylie Nord, Jeff Liang, John Hoffman, Jonas Kohler, Kaolin Fire, Karthik Sivakumar, Lawrence Chen, Licheng Yu, Luya Gao, Markos Georgopoulos, Rashel Moritz, Sara K. Sampson, Shikai Li, Simone Parmeggiani, Steve Fine, Tara Fowler, Vladan Petrovic, and Yuming Du. Movie gen: A cast of media foundation models, 2024. 2

[38] Ethan Pronovost, Meghana Reddy Ganesina, Noureldin Hendy, Zeyu Wang, Andres Morales, Kai Wang, and Nick Roy. Scenario diffusion: Controllable driving scenario generation with diffusion. In *NeurIPS*, 2023. 2, 3

[39] Ethan Pronovost, Kai Wang, and Nick Roy. Generating driving scenes with diffusion. In *ICRA Workshop on Scalable Autonomous Driving*, 2023. 2

[40] Runway ML. Introducing general world models, 2024. Accessed on 2024-10-24. 2

[41] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *ICLR*, 2022. 4

[42] Shuo Sun, Zekai Gu, Tianchen Sun, Jiawei Sun, Chengran Yuan, Yuhang Han, Dongen Li, and Marcelo H Ang. Drivescenegen: Generating diverse and realistic driving scenarios from scratch. *IEEE Robotics and Automation Letters*, 2024. 2, 3

[43] Simon Suo, Sebastian Regalado, Sergio Casas, and Raquel Urtasun. Trafficsim: Learning to simulate realistic multi-agent behaviors. In *CVPR*, 2021. 3

[44] Shuhan Tan, Kelvin Wong, Shenlong Wang, Sivabalan Manivasagam, Mengye Ren, and Raquel Urtasun. Scenegen: Learning to generate realistic traffic scenes. In *CVPR*, 2021. 3

[45] Shuhan Tan, Boris Ivanovic, Xinshuo Weng, Marco Pavone, and Philipp Krähenbühl. Language conditioned traffic generation. *7th Annual Conference on Robot Learning (CoRL)*, 2023. 3

[46] Shuhan Tan, Boris Ivanovic, Yuxiao Chen, Boyi Li, Xinshuo Weng, Yulong Cao, Philipp Krähenbühl, and Marco Pavone. Promptable closed-loop traffic simulation. *8th Annual Conference on Robot Learning (CoRL)*, 2024. 3

[47] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P. Srinivasan, Jonathan T. Barron, and Henrik Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *CVPR*, 2022. 3

[48] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 62(2):1805, 2000. 5

[49] Dani Valevski, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter. Diffusion models are real-time game engines. In *ICLR*, 2025. 2

[50] Hua Wei, Guanjie Zheng, Vikash Gayah, and Zhenhui Li. A survey on traffic signal control methods. *arXiv preprint arXiv:1904.08117*, 2019. 3

[51] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks 2021)*, 2021. 2

[52] Wei Wu, Xiaoxin Feng, Ziyan Gao, and Yuheng Kan. Smart: Scalable multi-agent real-time motion generation via next-token prediction. In *NeurIPS*, pages 114048–114071, 2024. 2

[53] Chen Yang, Aaron Xuxiang Tian, Dong Chen, Tianyu Shi, and Arsalan Heydarian. Wcdt: World-centric diffusion transformer for traffic scene generation, 2024. 2

[54] 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 *CVPR*, 2023. 3

[55] Chris Zhang, Sourav Biswas, Kelvin Wong, Kion Fallah, Lunjun Zhang, Dian Chen, Sergio Casas, and Raquel Urtasun. Learning to drive via asymmetric self-play. In *ECCV*, 2024. 3

[56] Zihan Zhang, Richard Liu, Rana Hanocka, and Kfir Aberman. Tedi: Temporally-entangled diffusion for long-term motion synthesis. In *ACM SIGGRAPH 2024 Conference Papers*, 2024. 2

[57] Ziyuan Zhong, Davis Rempe, Yuxiao Chen, Boris Ivanovic, Yulong Cao, Danfei Xu, Marco Pavone, and Baishakhi Ray. Language-guided traffic simulation via scene-level diffusion. In *CoRL*, 2023. 2

[58] Xiaoyu Zhou, Zhiwei Lin, Xiaojun Shan, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. Drivinggaussian: Composite gaussian splatting for surrounding dynamic autonomous driving scenes. In *CVPR*, 2024. 3