Scenario Dreamer: Vectorized Latent Diffusion for Generating Driving Simulation Environments

Luke Rowe [1,2,6], Roger Girgis[1,3,6], Anthony Gosselin[1,3],
Liam Paull[1,2,5], Christopher Pal[1,2,3,5], Felix Heide[4,6]
[1]Mila, [2]Université de Montréal, [3]Polytechnique Montréal,
[4]Princeton University, [5]CIFAR AI Chair, [6]Torc Robotics
https://princeton-computational-imaging.github.io/scenario-dreamer

Abstract

We introduce Scenario Dreamer, a fully data-driven generative simulator for autonomous vehicle planning that generates both the initial traffic scene—comprising a lane graph and agent bounding boxes—and closed-loop agent behaviours. Existing methods for generating driving simulation environments encode the initial traffic scene as a rasterized image and, as such, require parameter-heavy networks that perform unnecessary computation due to many empty pixels in the rasterized scene. Moreover, we find that existing methods that employ rule-based agent behaviours lack diversity and realism. Scenario Dreamer instead employs a novel vectorized latent diffusion model for initial scene generation that directly operates on the vectorized scene elements and an autoregressive Transformer for data-driven agent behaviour simulation. Scenario Dreamer additionally supports scene extrapolation via diffusion inpainting, enabling the generation of unbounded simulation environments. Extensive experiments show that Scenario Dreamer outperforms existing generative simulators in realism and efficiency: the vectorized scene-generation base model achieves superior generation quality with around 2× fewer parameters, 6× lower generation latency, and 10× fewer GPU training hours compared to the strongest baseline. We confirm its practical utility by showing that reinforcement learning planning agents are more challenged in Scenario Dreamer environments than traditional non-generative simulation environments, especially on long and adversarial driving environments.

1. Introduction

Simulators are invaluable tools for the safe and scalable development of autonomous vehicles (AVs) [6, 38, 54, 75]. They reduce costs and accelerate development, while offering a safe way to evaluate the performance of autonomous vehicle systems under diverse, rare, and potentially dangerous real-world conditions. Despite significant advancements in enhancing the speed and realism of autononous driving simulators [19, 30, 31], current simulators remain constrained by their reliance on prerecorded driving logs. Specifically, current data-driven simulators either replay or modify existing driving logs, limiting scalability due to the finite size and diversity of the available data. For instance, public driving simulators like GPUDrive [31] and Waymax [19], which are based on the Waymo Open Motion Dataset [13], cover only 1,750 km of unique roadway—a fraction of the nearly 22,000 km of roadway that an average human drives annually [15]. Simulations are also bounded by the length of the driving logs, which are typically less than 30 seconds. Given the billions of miles of diverse driving experience needed to validate the safety of autonomous vehicles in simulation, existing simulation environments derived from traffic logs have struggled to enable scalable training and evaluation of autonomous vehicles.

To address the limitations of current driving simulators, recent work has shown that diffusion generative models can be used to synthesize driving environments in an abstract representation comprising bird's eye view (BEV) agent bounding boxes and a lane graph [10, 56]. However, these methods are either not fully data-driven as the agent behaviours are governed by unrealistic rule-based models [10], or can only generate short driving scenarios with non-reactive agents [56]. Prior methods also encode driving scenes as rasterized BEV images, which require costly and unnecessary computation due to many empty pixels in the rasterized scenes [10, 56]. While existing generative simulators from sensor data have been proposed [74], these methods fail to meet the strict latency demands of simulators due to their reliance on large video diffusion models.

We propose Scenario Dreamer, a fully data-driven *generative* simulator for autonomous driving planning. Scenario Dreamer decomposes environment generation into

initial scene generation, which generates the initial scene in an abstract representation comprising BEV agent bounding boxes and a lane graph, and *behaviour simulation*, which controls agent behaviours over time. A core component of Scenario Dreamer is our proposed vectorized latent diffusion model for initial scene generation that *operates directly on the vectorized scene elements.* Compared to a rasterized encoding, the proposed model offers several practical advantages, including higher fidelity generations with reduced generation latency and higher quality scene extrapolations even at dense and complicated road geometries. It additionally enables learning the lane graph connectivity explicitly by modeling pairwise relationships between road elements, thus eliminating the need for post-processing heuristics used by Chitta et al. [10]. For closed-loop behaviour simulation, we utilize a return-conditioned autoregressive Transformer [53] for controllable agent behaviours that is adapted to support longer simulation rollouts. Both models are trained from real-world driving data.

Unlike prior work, Scenario Dreamer supports flexible test-time control over the generation of challenging and safety-critical scenarios. Specifically, we can explicitly control scene density with Scenario Dreamer by specifying the number of lanes and agents within a given field of view (FOV). Moreover, we can simulate adversarial agent behaviours with Scenario Dreamer via exponential tilting of the behaviour model [35, 53]. We integrate Scenario Dreamer with GPUDrive [31] to enable training Scenario Dreamer compatible reinforcement learning (RL) agents and evaluate RL agents in Scenario Dreamer environments. We show that Scenario Dreamer provides more challenging environments for RL planning agents than existing non-generative environments. Scenario Dreamer unlocks the ability to synthesize a limitless quantity of unbounded and interactive safety-critical environments, which we hope stimulates future autonomous driving planning research.

Contributions: (1) We introduce Scenario Dreamer, a fully data-driven *generative* simulator for autonomous driving planning. At the core of Scenario Dreamer is a novel vectorized latent diffusion model for initial scene generation that offers practical advantages over prior methods that utilize a rasterized scene encoding. (2) We show that Scenario Dreamer environments challenge RL planners, especially on long and adversarial driving environments.

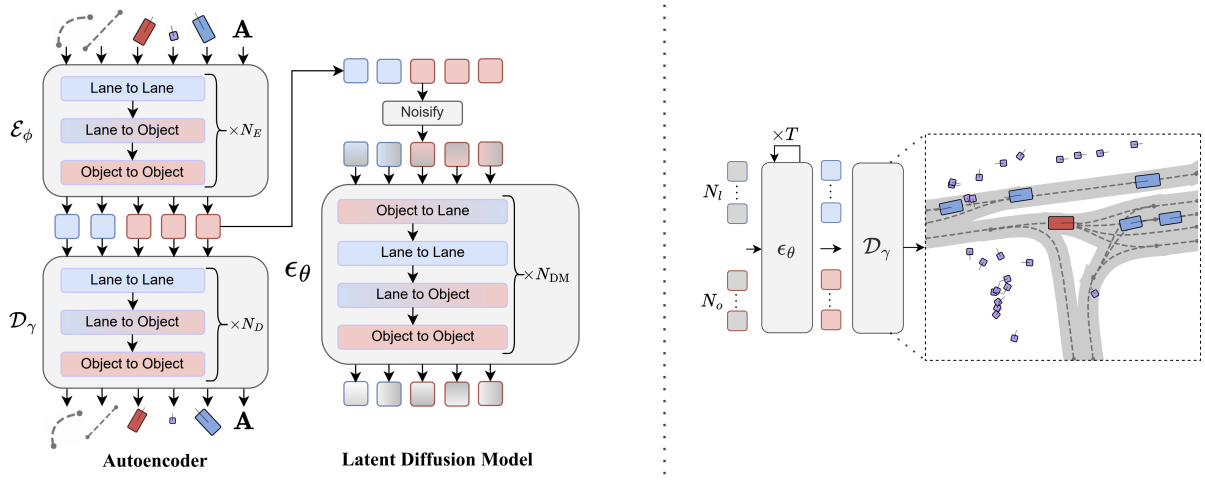
2. Related Work

Simulators for Autonomous Driving Traditional handcrafted driving simulators require costly manual effort to create environments [5, 12, 36]. Consequently, they are difficult to scale to larger environments and suffer from a significant sim-to-real gap. To address these issues, recent works have proposed data-driven driving simulators that are based on real driving data [6]. These simulators either reconstruct driving environments from sensor inputs using neural rendering techniques [9, 42, 47, 61, 68, 71, 72, 75, 80, 81, 83], or they utilize post-perception outputs to represent driving scenes in an abstract representation comprising agent bounding boxes and HD maps [19, 30, 31, 34, 37, 65, 82]. Scenario Dreamer adopts the abstract environment representation due to its lower computational demands and memory footprint, while offering sufficient complexity to evaluate AV planners.

Generative Simulation Environments While open-source data-driven simulators reduce the sim-to-real gap compared to handcrafted simulators, they are fundamentally limited by the scarcity of public driving data and the short durations of prerecorded driving logs. To overcome these constraints, recent works have proposed using diffusion generative models to generate driving simulation environments from sensor inputs [17, 23, 74] or post-perception data [10, 56]. A similar line of work uses diffusion models as generative interactive environments in other contexts, such as for video games [2, 4, 63] and robotics [66, 73, 76]. SLEDGE employs a customized latent diffusion model to generate bounding boxes and lane graphs but relies on rule-based agent behaviours [32, 62], limiting realism [10]. DriveSceneGen uses a diffusion model for initial scene generation and a motion prediction model for agent behaviours but produces only short, non-reactive scenarios of up to 20 seconds [56]. Scenario Dreamer addresses these shortcomings by introducing a latent diffusion model for initial scene generation and a Transformer-based driving policy [53] for closed-loop behaviour simulation, making it a fully data-driven, closed-loop, generative driving simulator capable of generating arbitrarily long simulations.

Initial Scene Generation and Behaviour Simulation A large body of work focuses specifically on the task of initial scene generation, targeting either the generation of initial agent bounding boxes [43, 59], lane graphs [45], or both [10, 56]. SceneControl [43] utilizes a vectorized diffusion model for initial agent generation but does so in data space. Both SLEDGE and DriveSceneGen employ diffusion models to generate the initial scene by processing it as a rasterized image, leading to unnecessary computation. In contrast, Scenario Dreamer generates the initial scene with vectorized processing [7, 8, 16, 18, 22, 27, 28, 40, 41, 46, 52, 70, 84], which enhances the speed and performance of the scene generations and extrapolations.

Recent works have proposed various methods for simulating closed-loop agent behaviours conditioned on initial agent states and HD maps [11, 26, 57, 58, 69, 77, 85]. These approaches rely on large Transformer-based models [24, 49, 67] or diffusion models [20, 25, 29, 78, 79], which are computationally expensive, especially considering that behaviour models run autoregressively in the loop during simulation. Scenario Dreamer utilizes a lightweight



(a) **Model Overview** (b) **Inference Time Sampling**

Figure 1. **Scenario Dreamer vectorized latent diffusion model for initial scene generation.** *Left*: We embed each vectorized scene element into a latent representation with an autoencoder parameterized with factorized attention blocks, which additionally fuses the lane connectivity $\mathbf{A}$. The latent Transformer diffusion model ϵ_θ is trained to sample from the autoencoder's latent distribution. *Right*: Scenario Dreamer samples novel driving scenes by initializing $N_o + N_l$ noise tokens which are iteratively denoised with ϵ_θ over T steps and decoded into vectorized scene elements. The ego vehicle is denoted in red, with other agents colored in blue and pedestrians in purple.

return-conditioned behaviour model [53] for closed-loop behaviour simulation. The return conditioning allows for flexible test-time control over agent behaviours, which can be used to generate adversarial driving behaviours. Researchers have also explored methods that attempt to generate both initial agent states and behaviours conditioned on the map and ego state, either by using separate modules for each task [3, 14] or by coupling them in an end-to-end trainable system [29, 44, 50, 60]. Scenario Dreamer keeps initial scene generation and behaviour simulation decoupled by design, allowing for these components to be invoked at different frequencies as needed.

3. Scenario Dreamer

We approach the task of generative driving simulation by decomposing it into *initial scene generation* and *behaviour simulation*. Below, we detail our proposed initial scene generator and agent behaviour model, both trained from real-world driving data. We then describe our generative simulation framework that supports closed-loop evaluation of AV planning agents.

3.1. Problem Setting

Initial Scene Generation Initial scene generation involves generating the initial BEV object bounding box states and the underlying map structure within a fixed field of view (FOV). Following [10], we generate a 64m×64m FOV centered and rotated to the ego agent. We denote by F the 64m×64m region being generated and denote by F_P and F_N the 32m × 64m regions that are ahead of and behind the ego agent, respectively. The initial scene generator is tasked with sampling from the distribution $p(\mathcal{I}_F)$ of initial scenes, where an initial scene $\mathcal{I}_F = \{\mathcal{O}, \mathcal{M}\}$ comprises a set of objects $\mathcal{O}$ and the map structure $\mathcal{M}$ within the FOV F. We define $\mathcal{O} = \{\mathbf{o}_i\}_{i=1}^{N_o}$ as a set of N_o objects that includes the traffic participants (*e.g.,* ego agent, vehicles, pedestrians, cyclists) and static objects (*e.g.,* traffic cones), where $\mathbf{o}_i$ is an 8-dimensional vector containing the 2-dimensional position, speed, cosine and sine of the heading, length, width, and object class of bounding box i. We generate a map representation $\mathcal{M}$ similar to [10], where $\mathcal{M} = \{\mathcal{L}, \mathbf{A}\}$ contains a set $\mathcal{L} = \{\mathbf{l}_i\}_{i=1}^{N_l}$ of N_l centerlines where each $\mathbf{l}_i$ is a 20×2 sequence of centerline positions, and $\mathbf{A} \in \{0,1\}^{N_l \times N_l \times 4}$ defines the associated centerline connectivity as a stack of four adjacency matrices describing the successor, predecessor, left, and right neighbor connections. The initial scene generator must also support sampling from the conditional distribution $p(\mathcal{I}_{F_P} | \mathcal{I}_{F_N})$ as this enables sampling arbitrarily long scenes by stitching together new generated regions F_P conditioned on existing regions F_N (see Figure 3).

Behaviour Simulation Given an initial scene configuration prescribed by the initial scene generator, the task of behaviour simulation involves modeling the behaviour of the dynamic objects in the scene over time. Concretely, given the set of initial object bounding box states $\mathbb{S}_0 := \mathcal{O}$ and the lane structure $\mathcal{M}$, the behaviour model employs a multi-agent driving policy $\pi(\mathbb{A}_t | \mathbb{S}_t, \mathcal{M})$ and a forward transition model $\mathcal{P}(\mathbb{S}_{t+1} | \mathbb{S}_t, \mathbb{A}_t)$, where $\mathbb{A}_t$ is set of the actions of all dynamic objects at timestep t.

3.2. Vectorized Latent Diffusion Model

The initial scene generator aims to sample from $p(\mathcal{I}_F)$. To achieve this, we propose learning an approximation, $\tilde{p}(\mathcal{I}_F) \approx p(\mathcal{I}_F)$, based on real driving data with a diffusion model, given the strong capability of diffusion models

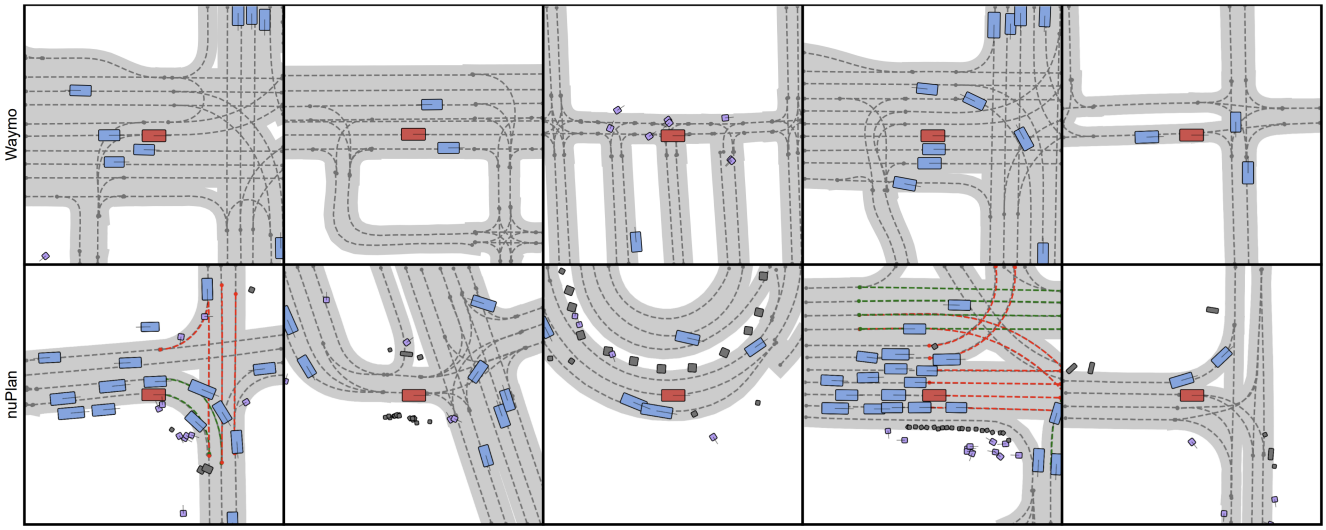


Figure 2. Vectorized environments generated by Scenario Dreamer with the proposed vectorized latent diffusion model trained on the Waymo dataset (top row) and nuPlan dataset (bottom row).

to capture highly complex distributions [1, 51]. We employ a two-stage training process: first, we learn a compact latent representation of the N_o objects, $\{\mathbf{h}_i^{\mathcal{O}}\}_{i=1}^{N_o}$, and the N_l centerlines, $\{\mathbf{h}_i^{\mathcal{L}}\}_{i=1}^{N_l}$, through a low-$\beta$ variational autoencoder [21]; next, we train a diffusion model to sample from the latent distribution $p(H) := p(\{\mathbf{h}_i^{\mathcal{O}}\}_{i=1}^{N_o}, \{\mathbf{h}_i^{\mathcal{L}}\}_{i=1}^{N_l})$. The architecture is illustrated in Figure 1, and detailed below.

3.2.1 Autoencoder

The autoencoder consists of a Transformer-based [64] encoder $\mathcal{E}_\phi$ and decoder $\mathcal{D}_\gamma$ that operate directly on the vectorized scene elements – in contrast, prior work [10] encodes the driving scene as a rasterized image.

Encoder The encoder first embeds the $N_o + N_l$ vector elements with a per-vector MLP and additionally embeds the one-hot representation of the lane connectivity type c_{ij} with an MLP for all centerline segment pairs i and j. $\mathcal{E}_\phi$ then applies a sequence of N_E factorized attention blocks over the $N_o + N_l$ embedded scene elements, where each factorized attention block comprises a lane-to-lane, lane-to-object, and object-to-object multi-head attention layer [18, 39, 46]. Lane-to-lane attention captures spatial relations between centerline segments, where the embedded lane connectivities are additionally fused into the attention keys and values. Lane-to-object attention incorporates map context into object embeddings, while object-to-object attention captures spatial relationships among objects. Following the N_E factorized attention blocks, the encoder maps each object and lane embedding to latent dimensions K_o and K_l respectively, with both mean and variance parameterized as in a VAE [33]. Importantly, we design $\mathcal{E}_\phi$ such that lane latents do not depend on the object features (*i.e.,* no object-to-lane attention), to allow lane-conditioned object generation at inference.

Decoder The decoder $\mathcal{D}\gamma$ samples from the latent distribution parameterized by the encoder $\{\{\mathbf{h}_i^{\mathcal{O}}\}_{i=1}^{N_o}, \{\mathbf{h}_i^{\mathcal{L}}\}_{i=1}^{N_l}\} \sim \mathcal{E}\phi$ and processes the embedded lane and object latents through a sequence of N_D factorized attention blocks. Following these blocks, the decoder reconstructs the continuous lane and object vector inputs, supervised by an ℓ_2 loss. For lane connectivity prediction, each pair of lane embeddings is concatenated and passed through an MLP to predict a categorical distribution over the connectivity type, trained with a cross-entropy loss. The Scenario Dreamer autoencoder is trained with the standard Evidence Lower Bound (ELBO) objective with low-β regularization, as detailed in the Appendix.

3.2.2 Latent Diffusion Sampling

We train a latent diffusion model to sample from the autoencoder's latent distribution $p(H)$ factorized as

$$p(H) = \sum_{N_o, N_l} p(\{\mathbf{h}_i^{\mathcal{O}}\}_{i=1}^{N_o}, \{\mathbf{h}_i^{\mathcal{L}}\}_{i=1}^{N_l} | N_o, N_l) p(N_o, N_l).$$

$p(N_o, N_l)$ is approximated with training set statistics, and thus we approximate $p(H)$ as $p(H) \approx p_\theta(H) = \sum_{N_o, N_l} p_\theta(\{\{\mathbf{h}_i^{\mathcal{O}}\}_{i=1}^{N_o}, \{\mathbf{h}_i^{\mathcal{L}}\}_{i=1}^{N_l}\} | N_o, N_l) p(N_o, N_l)$. Here, the conditional distribution $p_\theta(\cdot | N_o, N_l)$ is parameterized with a diffusion model with weights θ, which samples N_o object latents and N_l lane latents. Unlike image-based models, p_θ must accommodate variable latent sizes, so we design a customized transformer architecture with AdaLN-Zero conditioning [48]. This architecture, similar to the autoencoder, is composed of a sequence of N_{DM} *factorized* attention blocks (illustrated in Figure 1) that include sequential object-to-lane, lane-to-lane, lane-to-object, and object-to-object attention layers. This factorized processing approach allows each layer to model the layer-specific interactions while allowing for different hidden dimensions for object and lane tokens. Notably,

lane tokens require a larger hidden dimension due to the high level of spatial reasoning and detail required for realistic lane generation, while object tokens are effectively represented with a smaller dimension, making the diffusion model more efficient overall. We employ the standard DDPM objective to train p_θ, where p_θ is parameterized as a noise-prediction network ϵ_θ that learns to predict the noise of noised lane and object latents at varying noise levels:

$$L_{\text{dm}} = \mathbb{E}_{\mathbf{H}_t, \boldsymbol{\epsilon}_t \sim \mathcal{N}(0,1), t}\Big[\parallel \boldsymbol{\epsilon}_t - \epsilon_\theta(\mathbf{H}_t, t) \parallel_2^2 \Big],$$

where $\mathbf{H}_t$ denotes a stacked tensor of N_o object latents and N_l lane latents that is noised over t forward diffusion steps with noise vector $\boldsymbol{\epsilon}_t := (\boldsymbol{\epsilon}_\mathcal{L}, \boldsymbol{\epsilon}_\mathcal{O})$ containing both the object and lane noise vectors. We refer readers to the Appendix for more details.

Permutation Ambiguity Unlike images, which have a natural grid-like structure, set-structured data in Scenario Dreamer poses unique challenges for diffusion models. Specifically, a phenomenon called *permutation ambiguity* [7] arises: the vector elements that are sufficiently noised during training lose enough underlying structure that the model cannot reliably infer the permutation of the ground-truth signal toward which it should regress. In image-based transformer diffusion methods, such as DiT, this issue is resolved by applying a positional encoding to each grid patch. To similarly address permutation ambiguity in our model, we introduce sinusoidal positional encodings to the latent tokens prior to the factorized attention blocks.

Our positional encoding scheme involves defining an *ordering* of the tokens (per token type) during training, allowing the model to better infer a noised token's likely relative spatial position and thus the ground-truth signal toward which it should regress. To achieve this, we propose a recursive ordering procedure (see Appendix): tokens are ordered by minimum x-value, and if x-values differ by less than ϵ meters, they are subsequently ordered by minimum y-value, then maximum x-value, and finally maximum y-value. Notably, the initial ordering by x-value facilitates inpainting, as tokens in $\mathcal{I}_{F_N}$ form a contiguous subsequence of ordered tokens that precede those in $\mathcal{I}_{F_P}$.

3.2.3 Scene Generation

The Scenario Dreamer latent diffusion model supports multiple modes of scene generation within a single trained model: *initial scene generation* to sample 64m×64m scenes from $p(\mathcal{I}_F)$; *lane-conditioned object generation* to sample object bounding boxes $\mathcal{O}$ conditioned on a known map $\mathcal{M}$; and *scene inpainting* to sample from $p(\mathcal{I}_{F_P}|\mathcal{I}_{F_N})$. We describe each generation mode in detail in the following.

Initial Scene Generation. Scenario Dreamer generates novel initial driving scenes $\mathcal{I}_F$ by first sampling $(N_o, N_l) \sim p(N_o, N_l)$ from the joint distribution found in the training data, with dataset-dependent limits on N_o and N_l to ensure realistic scene density. Alternatively, users may directly specify (N_o, N_l), which enables control over the scene density. After setting (N_o, N_l), we sample latents $\{\mathbf{h}_i^\mathcal{O}\}_{i=1}^{N_o}, \{\mathbf{h}_i^\mathcal{L}\}_{i=1}^{N_l}$ from the diffusion model over $T = 100$ diffusion steps and decode the latents into vectorized scene elements using the autoencoder decoder.

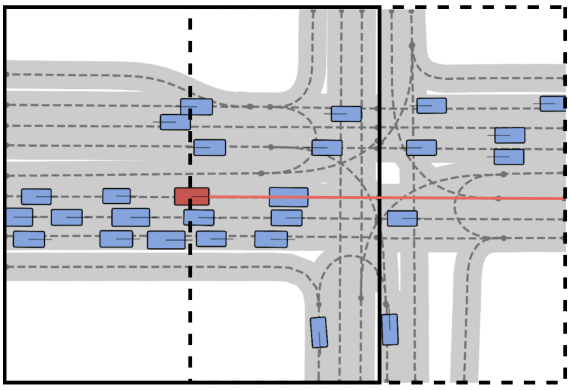


Figure 3. Illustrative example of Scenario Dreamer's inpainting capabilities, where the initial tile is outlined in solid lines and the inpainted tile in dashed. The model generates consistent lane geometries at the scene boundaries, even at complex intersections.

Lane-conditioned Object Generation involves encoding the vectorized elements of a given map $\mathcal{M}$ using $\mathcal{E}_\phi$, then sampling $N_o \sim p(N_o|N_l)$ object latents from the diffusion model, which are diffused while conditioning on the encoded map latents at each denoising timestep. The resulting object latents can be decoded into object bounding box configurations with D_γ.

Scene Inpainting. Following [10], we frame conditional sampling from $P(\mathcal{I}_{F_P}|\mathcal{I}_{F_N})$ as an inpainting task. Unlike grid-structured images, where boundaries between regions such as $\mathcal{I}_{F_P}$ and $\mathcal{I}_{F_N}$ are naturally defined (e.g., along $x = 0$), lane vectors in our setting can cross this boundary without restriction. To address this, we preprocess our dataset into two scene types: *partitioned* scenes, which are artificially split at $x = 0$, and *non-partitioned* scenes, where lane vectors may span across $x = 0$. The autoencoder is trained to reconstruct both scene types, while the diffusion model generates both by using a conditioning label to distinguish between them. To further enhance inpainting quality, during training the diffusion model conditions on the encoded latents of $\mathcal{I}_{F_N}$ for partitioned scenes, thus training it explicitly to inpaint. This significantly improves inpainting performance, as the model can learn to utilize the relevant scene context from $\mathcal{I}_{F_N}$ to ensure that the lane geometries remain spatially consistent across the $x = 0$ boundary.

Scenario Dreamer requires specifying the number of new lane and object vectors to occupy the new 32x64 region $\mathcal{I}_{F_P}$, effectively sampling from

$p(N_o^{F_P}, N_l^{F_P} | \mathcal{I}_{F_N})$. Although approximate sampling from $p(N_o^{F_P}, N_l^{F_P} | N_o^{F_N}, N_l^{F_N})$ is possible through training statistics, $p(N_l^{F_P} | \mathcal{I}^{\mathcal{F}_\mathcal{N}})$ is highly geometry-dependent. To account for this, we train a classifier, $f_\phi(N_l^{F_P} | \mathcal{M}_{F_N})$, alongside $\mathcal{E}_\phi$ to predict the number of lanes in $\mathcal{I}_{F_P}$ based on the context $\mathcal{I}_{F_N}$, trained only on partitioned scenes. Specifically, a learnable query vector cross-attends with the lane tokens in $\mathcal{I}_{F_N}$ within each factorized attention block of $\mathcal{E}_\phi$, outputting a categorical distribution over $N_o^{F_P}$ trained with a cross-entropy loss. At inference, we first sample $N_l^{F_P} \sim f_\phi$, followed by sampling $N_o^{F_P} \sim p(N_o^{F_P} | N_o^{F_N}, N_l^{F_N} + N_l^{F_P})$. With $(N_o^{F_P}, N_l^{F_P})$ sampled, we encode the latents for $\mathcal{I}_{F_N}$, and we generate $N_o^{F_P}$ and $N_l^{F_P}$ new tokens initialized to Gaussian noise. Standard diffusion inpainting [10] is then applied, where noised tokens in $\mathcal{I}_{F_N}$ are set to their encoded latents at each denoising step. The resulting latents are decoded to produce the new scene elements in $\mathcal{I}_{F_P}$.

3.3. Behaviour Simulation

Starting with an initial scene $\mathcal{I}$ generated by the latent diffusion model, we extend CtRL-Sim [53], an autoregressive Transformer-based behaviour model, to control multiple agent types (e.g., vehicles, pedestrians, and cyclists). To support multiple agents types, we use the k-disks tokenization scheme Philion et al. [49]. CtRL-Sim is a return-conditioned multi-agent policy designed for behaviour simulation. We adapt the CtRL-Sim architecture, which parameterizes the joint distribution over future returns $\mathbb{G}_t$ and actions $\mathbb{A}_t$, decomposed as $p_\theta(\mathbb{A}_t, \mathbb{G}_t | \mathbb{S}_t) = \pi_\theta(\mathbb{A}_t | \mathbb{S}_t, \mathbb{G}_t) p_\theta(\mathbb{G}_t | \mathbb{S}_t)$. A key advantage of this decomposition is its return-conditioning, which enables exponential tilting [35] of the learned return model at inference to generate good or adversarial driving behaviours. In the Scenario Dreamer framework, we aim to create adversarial scenarios that specifically challenge the autonomous vehicle (AV) planner. To this end, we design a reward function that penalizes collisions with the ego vehicle, and model the discounted return $G_t = \sum_{t=t}^{t+H} r_t$ based on the cumulative rewards over a horizon $H = 2$s, which we found offers improved controllability over modeling the full return.

3.4. Simulation Framework

With Scenario Dreamer in hand, we describe our proposed simulation framework's unique properties. First, Scenario Dreamer supports the evaluation of AV planners within its generative simulation environments over arbitrarily long simulation lengths. We define a route for the AV planner to follow, and simulate the other agents using CtRL-Sim. We follow the Waymo dataset filtering scheme employed in Nocturne [65] and GPUDrive [31] to ensure that the scenarios are valid for simulation (*i.e,* absence of traffic lights, which lack annotated traffic light states). To ensure that Scenario Dreamer-generated scenarios are valid for simulation, Scenario Dreamer conditions on a binary indicator during training that indicates whether the training scene passes the Nocturne filtering scheme. At inference, we utilize classifier guidance to sample from simulation-compatible scenes. The simulation framework is described in more detail in the Appendix.

4. Experiments

Datasets The Scenario Dreamer latent diffusion model is separately trained on both the Waymo Open Motion Dataset [13] and the nuPlan dataset [30]. For Waymo, our model captures vehicles, pedestrians, and cyclists, and we process the lane centerlines and their connectivity, excluding other map elements. For nuPlan, we model vehicles, pedestrians and static objects, and we process the lane centerlines, lane connectivity, and traffic light states. For both datasets, we use a 64m $\times$64m FOV centered on the ego vehicle and centerlines are processed using the compression algorithm from Chitta et al. [10], detailed in the Appendix.

Metrics To evaluate the quality of initial scene generations, we assess our methods on both lane graph generation and initial agent bounding box generation tasks. For lane graph generation, we report **Urban Planning** metrics, following prior works [10, 45, 56]. These metrics measure the distributional realism of the generated lane graph connectivity by computing Frechet distances on node features for nodes with degree $\neq 2$, referred to as *key points*. The following features are computed: *Connectivity* computes the degrees of all key points across the lane graphs; *Density* computes the number of key points in each lane graph; *Reach* computes the number of available paths from each keypoint to others; and *Convenience* computes the Dijsktra path lengths for all valid paths between key points. As in Chitta et al. [10], the Urban Planning metrics are scaled by suitable powers of 10 for readability. **Frechet Distance (FD)** measures the perceptual quality of the generated spatial lane positions by computing Frechet distances between the penultimate layer lane embeddings of a separately trained Scenario Dreamer autoencoder model. We additionally measure the longest **Route Length** by traversing the generated lane graph from the origin, and **Endpoint Distance**, which measures the average distance between the endpoint of lane i and starting point of lane j for all predicted successor edges (i, j) in the generated lane graphs. All lane graph generation metrics are computed on 50k real lane graphs and 50k ground-truth test lane graphs.

For initial agent generation, we focus on the distributional realism of vehicles, reporting Jensen-Shannon Divergence (**JSD**) metrics as in Lu et al. [43], including: *Nearest Dist.* between each vehicle and its neighbours, *Lateral Dev.* from the closest centerlines, *Angular Dev.* from the closest centerline, *Length* and *Width* of the vehicles, and *Speed* of

Dataset	Method	Per-Scene Gen. Time (s) ↓	Num. Parameters	GPUh	Perceptual Quality ↓ FD	Urban Planning ↓ Conn.	Dens.	Reach	Conve.	Route Length ↑ (m)	Endpoint Dist. ↓ (m)
nuPlan	SLEDGE (DiT-L) [10]	0.48	539M	96	1.89	2.34	2.43	0.70	2.44	35.34±8.63	0.47±0.32
	SLEDGE (DiT-XL) [10]	0.67	769M	960	1.44	1.67	1.74	0.51	1.68	35.83±8.35	0.42±0.29
	Scenario Dreamer (B)	**0.09**	377M	96	1.05	0.28	0.45	0.07	**0.14**	**37.04**±10.21	0.32±0.80
	Scenario Dreamer (L)	0.16	679M	256	**0.67**	**0.18**	**0.43**	**0.03**	0.33	36.87±10.37	**0.25**±0.71
Waymo	DriveSceneGen (GT Raster)* [56]	-	-	-	40.59	4.53	1.18	0.64	5.58	**41.61**±18.61	**0.01**±0.00
	Scenario Dreamer (B)	**0.08**	376M	96	1.61	0.17	1.05	0.56	3.81	38.23±12.78	0.32±0.90
	Scenario Dreamer (L)	0.16	678M	256	**1.38**	**0.03**	**0.95**	**0.28**	**2.05**	38.92±13.56	0.21±0.75

Table 1. Assessment of lane graph generation evaluated on the Waymo Open Motion and nuPlan test datasets. For each metric and dataset, the best method is **bolded**. * denotes privileged version of the method.

Dataset	Method	Distributional JSD ↓ Near. Dist.	Lat. Dev.	Ang. Dev.	Len.	Wid.	Speed	Collision Rate ↓ (%)
nuPlan	SLEDGE (DiT-L) [10]	0.53	0.63	3.60	11.84	10.16	0.46	22.3
	SLEDGE (DiT-XL) [10]	0.49	0.49	3.26	11.16	10.29	0.47	21.2
	Scenario Dreamer (B)	0.12	0.15	**0.17**	**0.22**	**0.14**	0.07	11.9
	Scenario Dreamer (L)	**0.09**	**0.11**	0.18	0.25	0.17	**0.06**	**9.3**
Waymo	DriveSceneGen (GT Raster)* [56]	0.63	1.01	2.43	58.86	54.51	18.70	**0.2**
	Scenario Dreamer (B)	0.07	0.07	**0.07**	**0.40**	**0.25**	**0.36**	5.4
	Scenario Dreamer (L)	**0.06**	**0.05**	**0.07**	0.42	**0.25**	0.38	4.8

Table 2. Assessment of initial agent bounding box generation evaluated on the Waymo Open Motion and nuPlan test datasets. * denotes privileged version of the method.

the vehicles. The JSD metrics are computed over 50k real and generated scenes and scaled by suitable powers of 10. We also report the **Collision Rate** of the agent bounding boxes. Behaviour simulation metrics are similar to that of the agent generation, with details in the Appendix.

Methods under Comparison We evaluate Scenario Dreamer against competitive diffusion-based methods for initial scene generation on both the nuPlan and Waymo datasets. For nuPlan, we compare with the SLEDGE DiT-L and DiT-XL models [10], retraining them following their open-source code. SLEDGE encodes driving scenes into a latent space using an image rasterization scheme, which is then modeled by a latent diffusion process and decoded in a vectorized format. On the Waymo dataset, we benchmark against DriveSceneGen [56]. However, in our experiments, the DriveSceneGen diffusion model did not converge, so we instead use a privileged version that provides an *upper bound* on DriveSceneGen's performance by rasterizing 50k ground-truth training scenes following their rasterization pipeline and decoding them with DriveSceneGen's vectorization post-processing. Notably, Scenario Dreamer is the only model that fully processes scene elements in a vectorized manner. For behaviour simulation, we compare CtRL-Sim against the rule-based IDM [62] and a data-driven baseline Trajeglish [49], with quantitative and qualitative results reported in the Appendix. The Scenario Dreamer latent diffusion model is available in two sizes: the base model (B) with 377M parameters, trained over 24 hours on 4 A100 GPUs, and the large model (L) with 679M parameters, trained over 32 hours on 8 A100 GPUs. Additional model details are provided in the Appendix.

Results Table 1 reports the lane graph generation results on the nuPlan and Waymo datasets, along with per-scene generation times, parameter counts, and diffusion model GPU training hours (GPUh) for each method. On the nuPlan dataset, Scenario Dreamer outperforms SLEDGE across all lane graph generation metrics. Notably, the smaller Scenario Dreamer base model surpasses SLEDGE's largest DiT-XL model on every metric, with approximately $2\times$ fewer parameters, $7\times$ lower inference latency, and $10\times$ fewer GPU training hours. We note that our reproduction of the SLEDGE DiT-XL model underperformed the results reported in Chitta et al. [10]. Nevertheless, Scenario Dreamer (L) still outperforms the published results in Chitta et al. [10] across all metrics. These results highlight the effectiveness of the vectorized design for lane graph generation.

The vectorized processing enhances efficiency by focusing parameters directly on the relevant vectorized scene elements rather than processing empty pixels and is not sensitive to the specific rasterization scheme that is employed. With vectorized processing, each latent corresponds to a scene element, allowing the model to effectively capture and learn the interactions between scene elements. Practically, this results in Scenario Dreamer's latent diffusion model being substantially more efficient than SLEDGE, as the vectorized latent representation requires far fewer Transformer layers to model effectively. Concretely, the largest SLEDGE diffusion model requires 28 Transformer layers for processing, whereas Scenario Dreamer's largest latent diffusion model requires only 6 lane to lane attention

Method	Per-Scene Gen. Time (s)	Perceptual Quality ↓ FD	Urban Planning ↓ Conn.	Dens.	Reach	Conve.	Route Length ↑ (m)	Endpoint Dist. ↓ (m)
Scenario Dreamer (Non-factorized)	0.20	1.21	0.51	0.67	0.16	0.55	36.71±9.95	0.37±0.76
Scenario Dreamer (No lane ordering)	**0.09**	1.36	0.60	**0.07**	0.21	0.71	36.70±9.94	0.35±0.74
Scenario Dreamer (Heuristic topology)	**0.09**	**1.05**	**0.28**	0.44	0.12	0.60	**37.13**±9.85	**0.30**±0.23
Scenario Dreamer	**0.09**	**1.05**	**0.28**	0.45	**0.07**	**0.14**	37.04±10.21	0.32±0.80

Table 3. Ablation experiments validating the design choices of Scenario Dreamer (B) for lane graph generation on the nuPlan dataset.

Other Agent Beh.	Test Env.	Avg. Route Len. (m)	Coll. (%)	Offroad (%)	Succ. (%)
Log Replay	Waymo Test	55	29.3±0.6	6.9±0.8	63.8±0.8
CtRL-Sim (Pos. Tilt)	Waymo Test	55	35.7±0.7	4.9±0.9	59.4±0.4
CtRL-Sim (Pos. Tilt)	SD (55m)	55	33.8±1.2	6.4±0.7	59.8±1.7
CtRL-Sim (Pos. Tilt)	SD (100m)	100	52.8±1.4	9.1±1.6	38.2±1.9
CtRL-Sim (Neg. Tilt)	SD (100m)	100	59.0±1.3	9.0±0.8	32.1±1.3

Table 4. **RL Planner Results.** PPO agents are trained in GPUDrive on 100 Waymo scenes and evaluated on 250 scenes (Waymo test scenes or 55m/100m-route Scenario Dreamer (SD) scenes). mean±std reported over 5 seeds.

layers. On the Waymo dataset, Scenario Dreamer outperforms the privileged version of the DriveSceneGen model on the Urban Planning and perceptual metrics. This advantage is attributed to substantial errors introduced by vectorizing a rasterized scene as a postprocessing step as done by DriveSceneGen, underscoring the importance of directly processing vectorized elements within the model itself.

Table 2 presents the results on agent generation on the nuPlan and Waymo datasets. Consistent with the lane graph generation results, the Scenario Dreamer model achieves more realistic agent configurations, as reflected by lower JSD metrics and fewer collisions compared to related methods. In the Appendix, we visualize a random sample of Scenario Dreamer, SLEDGE, and privileged DriveSceneGen generated scenes, where Scenario Dreamer scenes exhibit visually more realistic lane graph and agent configurations. We attribute these improvements to the model's vectorized scene processing approach.

Table 3 reports ablation experiments examining the impact of key architectural design choices. First, when we replace the learned lane connectivity prediction task with heuristic labels based on lane endpoint distances and relative orientations (*Heuristic topology*), Scenario Dreamer shows degraded performance on the Reach and Connectivity Urban Planning metrics, which best reflect the realism of lane graph connectivity. This result indicates that learning lane connectivity directly outperforms post-processing heuristics. Additionally, removing the proposed lane ordering (*No lane ordering*) introduces permutation ambiguity, which reduces the generation quality. Finally, we trained a non-factorized Scenario Dreamer diffusion model (*Non-factorized*) with a comparable parameter count and observed that factorized processing not only enhances performance but also reduces inference latency by approximately $2\times$, underscoring the effectiveness of the factorized design.

Table 4 summarizes the performance of a PPO [55] planner trained in GPUDrive [31] on 100 Waymo training scenes, reformatted to be compatible with Scenario Dreamer simulation environments. We evaluate this planner in both generative Scenario Dreamer environments with 55m *(SD (55m))* and 100m *(SD (100m))* routes and in non-generative Waymo test environments *(Waymo Test)*. For consistency, the Waymo maps are processed to retain only centerlines, ignoring other map elements. We measure collision rate (*Coll.*) with other agents, offroad rate (*Offroad*) based on a lateral deviation of more than 2.5m from the route, and success rate (*Succ.*) determined by whether the planner completes the route. The results indicate that replacing non-reactive log replay agent behaviors with reactive CtRL-Sim behaviors has only a marginal impact on the RL policy's performance (Row 1 vs. Row 2), underscoring the realism of CtRL-Sim. Moreover, evaluating the RL agents on Scenario Dreamer-generated environments with 55m routes—matching the average route length of the Waymo test scenes—yields comparable planner performance (Row 2 vs. Row 3), further affirming the fidelity of Scenario Dreamer. We show that increasing the route length to 100m notably degrades policy performance (Row 3 vs. Row 4), and negative tilting of CtRL-Sim further exacerbates this effect (Row 4 vs. Row 5), highlighting how longer and more adversarial scenarios challenge the RL planner.

5. Conclusion

We present Scenario Dreamer, a fully data-driven generative simulator for autonomous driving planners. At its core, Scenario Dreamer is comprised of an initial scene generator and a return-conditioned multi-agent Transformer behaviour model. A key novelty of this work is the vectorized latent diffusion model, enabling efficient and effective scene generation compared to rasterized scene encoding approaches. We hope that the Scenario Dreamer framework can be the foundation of future research on fully data-driven generative simulators for autonomous driving research and development.

Limitations We observed qualitatively that the traffic light signaling of lanes did not always provide valid traffic logic. Furthermore, Scenario Dreamer currently only generates centerline maps. In the future, we plan to generate other road element types (e.g., road edges, crosswalks).

6. Acknowledgements

Felix Heide was supported by an NSF CAREER Award (2047359), a Packard Foundation Fellowship, a Sloan Research Fellowship, a Sony Young Faculty Award, a Project X Innovation Award, and an Amazon Science Research Award. C. Pal thanks the IVADO, the NSERC Discovery Grants and the CIFAR AI Chairs programs for their support.

References

[1] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J. Ballard, Joshua Bambrick, Sebastian W. Bodenstein, David A. Evans, Chia-Chun Hung, Michael O'Neill, David Reiman, Kathryn Tunyasuvunakool, Zachary Wu, Akvilė Žemgulytė, Eirini Arvaniti, Charles Beattie, Ottavia Bertolli, Alex Bridgland, Alexey Cherepanov, Miles Congreve, Alexander I. Cowen-Rivers, Andrew Cowie, Michael Figurnov, Fabian B. Fuchs, Hannah Gladman, Rishub Jain, Yousuf A. Khan, Caroline M. R. Low, Kuba Perlin, Anna Potapenko, Pascal Savy, Sukhdeep Singh, Adrian Stecula, Ashok Thillaisundaram, Catherine Tong, Sergei Yakneen, Ellen D. Zhong, Michal Zielinski, Augustin Žídek, Victor Bapst, Pushmeet Kohli, Max Jaderberg, Demis Hassabis, and John M. Jumper. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, 2024. 4

[2] Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos J. Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari. *NeurIPS*, 2024. 2

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

[4] Jake Bruce, Michael D. Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Bechtle, Feryal M. P. Behbahani, Stephanie C. Y. Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott E. Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim Rocktäschel. Genie: Generative interactive environments. In *ICML*, 2024. 2

[5] Panpan Cai, Yiyuan Lee, Yuanfu Luo, and David Hsu. SUMMIT: A simulator for urban driving in massive mixed traffic. In *ICRA*, 2020. 2

[6] Di Chen, Meixin Zhu, Hao Yang, Xuesong Wang, and Yinhai Wang. Data-driven traffic simulation: A comprehensive review. *IEEE Trans. Intell. Veh.*, 2024. 1, 2

[7] Jiacheng Chen, Ruizhi Deng, and Yasutaka Furukawa. Polydiffuse: Polygonal shape reconstruction via guided set diffusion models. In *NeurIPS*, 2023. 2, 5

[8] Shaoyu Chen, Bo Jiang, Hao Gao, Bencheng Liao, Qing Xu, Qian Zhang, Chang Huang, Wenyu Liu, and Xinggang Wang. Vadv2: End-to-end vectorized autonomous driving via probabilistic planning. *arXiv*, 2024. 2

[9] Yurui Chen, Junge Zhang, Ziyang Xie, Wenye Li, Feihu Zhang, Jiachen Lu, and Li Zhang. S-nerf++: Autonomous driving simulation via neural reconstruction and generation. *IEEE TPAMI*, 2025. 2

[10] Kashyap Chitta, Daniel Dauner, and Andreas Geiger. SLEDGE: synthesizing driving environments with generative models and rule-based traffic. In *ECCV*, 2024. 1, 2, 3, 4, 5, 6, 7

[11] Daphne Cornelisse and Eugene Vinitsky. Human-compatible driving agents through data-regularized self-play reinforcement learning. *Reinforcement Learning Journal*, 1, 2024. 2

[12] Alexey Dosovitskiy, Germán Ros, Felipe Codevilla, Antonio M. López, and Vladlen Koltun. CARLA: an open urban driving simulator. In *CoRL*, 2017. 2

[13] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R. Qi, Yin Zhou, Zoey Yang, Aurelien 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. 1, 6

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

[15] FHWA. Average annual miles per driver by age group. `https://www.fhwa.dot.gov/ohim/onh00/bar8.htm`, 2022. Accessed: 2024-09-17. 1

[16] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding HD maps and agent dynamics from vectorized representation. In *CVPR*, 2020. 2

[17] Shenyuan Gao, Jiazhi Yang, Li Chen, Kashyap Chitta, Yihang Qiu, Andreas Geiger, Jun Zhang, and Hongyang Li. Vista: A generalizable driving world model with high fidelity and versatile controllability. In *NeurIPS*, 2024. 2

[18] Roger Girgis, Florian Golemo, Felipe Codevilla, Martin Weiss, Jim Aldon D'Souza, Samira Ebrahimi Kahou, Felix Heide, and Christopher Joseph Pal. Latent variable sequential set transformers for joint multi-agent motion prediction. In *ICLR*, 2022. 2, 4

[19] 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 *NeurIPS*, 2023. 1, 2

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

[21] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017. 4

[22] Emiel Hoogeboom, Victor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In *ICML*, 2022. 2

[23] 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*, 2023. 2

[24] Yihan Hu, Siqi Chai, Zhening Yang, Jingyu Qian, Kun Li, Wenxin Shao, Haichao Zhang, Wei Xu, and Qiang Liu. Solving motion planning tasks with a scalable generative model. In *ECCV*, 2024. 2

[25] Zhiyu Huang, Zixu Zhang, Ameya Vaidya, Yuxiao Chen, Chen Lv, and Jaime Fernández Fisac. Versatile behavior diffusion for generalized traffic agent simulation. *arXiv*, 2024. 2

[26] Maximilian Igl, Daewoo Kim, Alex Kuefler, Paul Mougin, Punit Shah, Kyriacos Shiarlis, Dragomir Anguelov, Mark Palatucci, Brandyn White, and Shimon Whiteson. Symphony: Learning realistic and diverse agents for autonomous driving simulation. In *ICRA*, 2022. 2

[27] Bo Jiang, Shaoyu Chen, Qing Xu, Bencheng Liao, Jiajie Chen, Helong Zhou, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. VAD: vectorized scene representation for efficient autonomous driving. In *ICCV*, 2023. 2

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

[29] Chiyu Max Jiang, Yijing Bai, Andre Cornman, Christopher Davis, Xiukun Huang, Hong Jeon, Sakshum Kulshrestha, John Wheatley 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

[30] 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. 1, 2, 6

[31] Saman Kazemkhani, Aarav Pandya, Daphne Cornelisse, Brennan Shacklett, and Eugene Vinitsky. GPUDrive: Data-driven, multi-agent driving simulation at 1 million FPS. In *ICLR*, 2025. 1, 2, 6, 8

[32] Arne Kesting, Martin Treiber, and Dirk Helbing. General lane-changing model mobil for car-following models. *Transportation Research Record*, 1999(1):86–94, 2007. 2

[33] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014. 4

[34] Jonathan Wilder Lavington, Ke Zhang, Vasileios Lioutas, Matthew Niedoba, Yunpeng Liu, Dylan Green, Saeid Naderiparizi, Xiaoxuan Liang, Setareh Dabiri, Adam Ścibior, Berend Zwartsenberg, and Frank Wood. Torch-driveenv: A reinforcement learning benchmark for autonomous driving with reactive, realistic, and diverse non-playable characters. *arXiv*, 2024. 2

[35] Kuang-Huei Lee, Ofir Nachum, Mengjiao Yang, Lisa Lee, Daniel Freeman, Sergio Guadarrama, Ian Fischer, Winnie Xu, Eric Jang, Henryk Michalewski, and Igor Mordatch. Multi-game decision transformers. In *NeurIPS*, 2022. 2, 6

[36] Edouard Leurent. An environment for autonomous driving decision-making. https://github.com/eleurent/highway-env, 2018. 2

[37] Quanyi Li, Zhenghao Peng, Lan Feng, Qihang Zhang, Zhenghai Xue, and Bolei Zhou. Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning. *IEEE TPAMI*, 2023. 2

[38] Yueyuan Li, Wei Yuan, Songan Zhang, Weihao Yan, Qiyuan Shen, Chunxiang Wang, and Ming Yang. Choose your simulator wisely: A review on open-source simulators for autonomous driving. *IEEE Trans. Intell. Veh.*, 2024. 1

[39] Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning lane graph representations for motion forecasting. In *ECCV*, 2020. 4

[40] Bencheng Liao, Shaoyu Chen, Xinggang Wang, Tianheng Cheng, Qian Zhang, Wenyu Liu, and Chang Huang. Maptr: Structured modeling and learning for online vectorized HD map construction. In *ICLR*, 2023. 2

[41] Bencheng Liao, Shaoyu Chen, Yunchi Zhang, Bo Jiang, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. Maptrv2: An end-to-end framework for online vectorized HD map construction. *Int. J. Comput. Vis.*, 2025. 2

[42] William Ljungbergh, Adam Tonderski, Joakim Johnander, Holger Caesar, Kalle Åström, Michael Felsberg, and Christoffer Petersson. Neuroncap: Photorealistic closed-loop safety testing for autonomous driving. *ECCV*, 2024. 2

[43] Jack Lu, Kelvin Wong, Chris Zhang, Simon Suo, and Raquel Urtasun. Scenecontrol: Diffusion for controllable traffic scene generation. In *ICRA*, 2024. 2, 6

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

[45] Lu Mi, Hang Zhao, Charlie Nash, Xiaohan Jin, Jiyang Gao, Chen Sun, Cordelia Schmid, Nir Shavit, Yuning Chai, and Dragomir Anguelov. Hdmapgen: A hierarchical graph generative model of high definition maps. In *CVPR*, 2021. 2, 6

[46] 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, Ben Sapp, Zhifeng Chen, and Jonathon Shlens. Scene transformer: A unified architecture for predicting future trajectories of multiple agents. In *ICLR*, 2022. 2, 4

[47] Julian Ost, Fahim Mannan, Nils Thuerey, Julian Knodt, and Felix Heide. Neural scene graphs for dynamic scenes. In *CVPR*, 2021. 2

[48] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*. IEEE, 2023. 4

[49] Jonah Philion, Xue Bin Peng, and Sanja Fidler. Trajeglish: Traffic modeling as next-token prediction. In *ICLR*, 2024. 2, 6, 7

[50] 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. 3

[51] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with CLIP latents. *arXiv*, 2022. 4

[52] Luke Rowe, Martin Ethier, Eli-Henry Dykhne, and Krzysztof Czarnecki. FJMP: factorized joint multi-agent motion prediction over learned directed acyclic interaction graphs. In *CVPR*, 2023. 2

[53] Luke Rowe, Roger Girgis, Anthony Gosselin, Bruno Carrez, Florian Golemo, Felix Heide, Liam Paull, and Christopher Pal. CtRL-sim: Reactive and controllable driving agents with offline reinforcement learning. In *CoRL*, 2024. 2, 3, 6

[54] John M Scanlon, Kristofer D Kusano, Tom Daniel, Christopher Alderson, Alexander Ogle, and Trent Victor. Waymo simulated driving behavior in reconstructed fatal crashes within an autonomous vehicle operating domain. *Accident Analysis & Prevention*, 2021. 1

[55] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv*, 2017. 8

[56] 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 RA-L*, 2024. 1, 2, 6, 7

[57] Simon Suo, Sebastian Regalado, Sergio Casas, and Raquel Urtasun. Trafficsim: Learning to simulate realistic multi-agent behaviors. In *CVPR*, 2021. 2

[58] Simon Suo, Kelvin Wong, Justin Xu, James Tu, Alexander Cui, Sergio Casas, and Raquel Urtasun. Mixsim: A hierarchical framework for mixed reality traffic simulation. In *CVPR*, 2023. 2

[59] Shuhan Tan, Kelvin Wong, Shenlong Wang, Sivabalan Manivasagam, Mengye Ren, and Raquel Urtasun. Scenegen: Learning to generate realistic traffic scenes. In *CVPR*, 2021. 2

[60] Shuhan Tan, Boris Ivanovic, Xinshuo Weng, Marco Pavone, and Philipp Kraehenbuehl. Language conditioned traffic generation. In *CoRL*, 2023. 3

[61] Adam Tonderski, Carl Lindström, Georg Hess, William Ljungbergh, Lennart Svensson, and Christoffer Petersson. Neurad: Neural rendering for autonomous driving. In *CVPR*, 2024. 2

[62] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 62(2):1805, 2000. 2, 7

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

[64] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 4

[65] Eugene Vinitsky, Nathan Lichtlé, Xiaomeng Yang, Brandon Amos, and Jakob Foerster. Nocturne: a scalable driving benchmark for bringing multi-agent learning one step closer to the real world. In *NeurIPS*, 2022. 2, 6

[66] Jialong Wu, Shaofeng Yin, Ningya Feng, Xu He, Dong Li, Jianye Hao, and Mingsheng Long. ivideogpt: Interactive videogpts are scalable world models. In *NeurIPS*, 2024. 2

[67] Wei Wu, Xiaoxin Feng, Ziyan Gao, and Yuheng Kan. SMART: scalable multi-agent real-time motion generation via next-token prediction. In *NeurIPS*, 2024. 2

[68] Ziyang Xie, Junge Zhang, Wenye Li, Feihu Zhang, and Li Zhang. S-nerf: Neural radiance fields for street views. In *ICLR*, 2023. 2

[69] Danfei Xu, Yuxiao Chen, Boris Ivanovic, and Marco Pavone. BITS: bi-level imitation for traffic simulation. In *ICRA*, 2023. 2

[70] Minkai Xu, Alexander S. Powers, Ron O. Dror, Stefano Ermon, and Jure Leskovec. Geometric latent diffusion models for 3d molecule generation. In *ICML*, 2023. 2

[71] Yunzhi Yan, Haotong Lin, Chenxu Zhou, Weijie Wang, Haiyang Sun, Kun Zhan, Xianpeng Lang, Xiaowei Zhou, and Sida Peng. Street gaussians for modeling dynamic urban scenes. In *ECCV*, 2024. 2

[72] Jiawei Yang, Boris Ivanovic, Or Litany, Xinshuo Weng, Seung Wook Kim, Boyi Li, Tong Che, Danfei Xu, Sanja Fidler, Marco Pavone, and Yue Wang. Emernerf: Emergent spatial-temporal scene decomposition via self-supervision. *ICLR*, 2024. 2

[73] Sherry Yang, Yilun Du, Seyed Kamyar Seyed Ghasemipour, Jonathan Tompson, Leslie Pack Kaelbling, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. In *ICLR*, 2024. 2

[74] Xuemeng Yang, Licheng Wen, Yukai Ma, Jianbiao Mei, Xin Li, Tiantian Wei, Wenjie Lei, Daocheng Fu, Pinlong Cai, Min Dou, Botian Shi, Liang He, Yong Liu, and Yu Qiao. Drivearena: A closed-loop generative simulation platform for autonomous driving. *arXiv*, 2024. 1, 2

[75] 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. 1, 2

[76] Youwei Yu, Junhong Xu, and Lantao Liu. Adaptive diffusion terrain generator for autonomous uneven terrain navigation. In *CoRL*, 2024. 2

[77] Zhejun Zhang, Alexander Liniger, Dengxin Dai, Fisher Yu, and Luc Van Gool. Trafficbots: Towards world models for autonomous driving simulation and motion prediction. In *ICRA*, 2023. 2

[78] 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

[79] Ziyuan Zhong, Davis Rempe, Danfei Xu, Yuxiao Chen, Sushant Veer, Tong Che, Baishakhi Ray, and Marco Pavone.

Guided conditional diffusion for controllable traffic simulation. In *ICRA*, 2023. 2

[80] Hongyu Zhou, Longzhong Lin, Jiabao Wang, Yichong Lu, Dongfeng Bai, Bingbing Liu, Yue Wang, Andreas Geiger, and Yiyi Liao. HUGSIM: A real-time, photo-realistic and closed-loop simulator for autonomous driving. *arXiv*, 2024. 2

[81] Hongyu Zhou, Jiahao Shao, Lu Xu, Dongfeng Bai, Weichao Qiu, Bingbing Liu, Yue Wang, Andreas Geiger, and Yiyi Liao. HUGS: holistic urban 3d scene understanding via gaussian splatting. In *CVPR*, 2024. 2

[82] Ming Zhou, Jun Luo, Julian Villella, Yaodong Yang, David Rusu, Jiayu Miao, Weinan Zhang, Montgomery Alban, Iman Fadakar, Zheng Chen, Aurora Chongxi Huang, Ying Wen, Kimia Hassanzadeh, Daniel Graves, Dong Chen, Zhengbang Zhu, Nhat Nguyen, Mohamed Elsayed, Kun Shao, Sanjeevan Ahilan, Baokuan Zhang, Jiannan Wu, Zhengang Fu, Kasra Rezaee, Peyman Yadmellat, Mohsen Rohani, Nicolas Perez Nieves, Yihan Ni, Seyedershad Banijamali, Alexander Cowen Rivers, Zheng Tian, Daniel Palenicek, Haitham bou Ammar, Hongbo Zhang, Wulong Liu, Jianye Hao, and Jun Wang. Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving, 2020. 2

[83] 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. 2

[84] Zikang Zhou, Luyao Ye, Jianping Wang, Kui Wu, and Kejie Lu. Hivt: Hierarchical vector transformer for multi-agent motion prediction. In *CVPR*, 2022. 2

[85] Zikang Zhou, Haibo Hu, Xinhong Chen, Jianping Wang, Nan Guan, Kui Wu, Yung-Hui Li, Yu-Kai Huang, and Chun Jason Xue. Behaviorgpt: Smart agent simulation for autonomous driving with next-patch prediction. In *NeurIPS*, 2024. 2