Solving Motion Planning Tasks with a Scalable Generative Model

Yihan Hu, Siqi Chai, Zhening Yang, Jingyu Qian, Kun Li, Wenxin Shao, Haichao Zhang, Wei Xu, and Qiang Liu

Horizon Robotics Inc.
{yihan.hu96, Proliu}@gmail.com

Abstract. As autonomous driving systems being deployed to millions of vehicles, there is a pressing need of improving the system's scalability, safety and reducing the engineering cost. A realistic, scalable, and practical simulator of the driving world is highly desired. In this paper, we present an efficient solution based on generative models which learns the dynamics of the driving scenes. With this model, we can not only simulate the diverse futures of a given driving scenario but also generate a variety of driving scenarios conditioned on various prompts. Our innovative design allows the model to operate in both full-Autoregressive and partial-Autoregressive modes, significantly improving inference and training speed without sacrificing generative capability. This efficiency makes it ideal for being used as an online reactive environment for reinforcement learning, an evaluator for planning policies, and a high-fidelity simulator for testing. We evaluated our model against two real-world datasets: the Waymo motion dataset and the nuPlan dataset. On the simulation realism and scene generation benchmark, our model achieves the state-of-the-art performance. And in the planning benchmarks, our planner outperforms the prior arts. We conclude that the proposed generative model may serve as a foundation for a variety of motion planning tasks, including data generation, simulation, planning, and online training. Source code is public at https://github.com/HorizonRobotics/GUMP/.

1 Introduction

Autonomous driving (AD) has evolved from a visionary concept to tangible products in recent years [49, 74, 79]. Despite these advancements, challenges in technical scalability remain [27]. These involve the system's ability to adapt to new and unseen environments, which is the root cause to ongoing safety concerns and the extensive engineering efforts required to address various failure scenarios. To continuously scale up in terms of safety and cost efficiency, developing a model that learns and represents the driving world is essential. Such a model could enable the continuous generation of data for training and testing, moving toward a more integrated and learned driving system that seamlessly interacts with real-world road users.

A driving world may be considered as a sequence of driving scenarios, which typically contain a map and multiple agents interacting within that space. Obtaining a local map for these scenarios is generally straightforward, whether through map providers [54] or through online mapping modules [43–45]. The real challenge, however, lies in understanding the dynamic nature of traffic - specifically, the interdependence among road users. Thus, a fundamental task in modeling the driving world involves accurately predicting the behaviors of these agents, formulating the plans based on these predictions and continuously refining these predictions as the agents take action.

This task has been intensively explored in recent years, including motion forecasting [21, 34, 51, 52, 67] and imitative traffic modeling [42, 71]. All of these works are able to model the either marginal or joint distributions of the multiagents' future trajectories. Despite these advancements, these models are typically trained in an open-loop setting, which may limit their ability to adapt to out-of-distribution states encountered in real-world, closed-loop settings. Consequently, their capability for traffic simulation is generally constrained to short duration, such as a few seconds.

Recently, closed-loop motion prediction has attracted increasing attentions, which addresses the problems of long horizon traffic simulation and interactive simulation [50, 56, 65, 77]. Through the iterative forecasting of each agent's next states, autoregressive (AR) models articulate the interactions among agents as a series of cascading conditional distributions. At each subsequent time step, sampling from the model can generate a variety of actions for an agent, aligning with both the context of the scene and the preceding actions of other agents. Therefore, learning the dynamics of the driving world is approached in the same manner as learning sequence prediction, akin to the techniques used in language modeling [60].

In light of these work, we propose the **G**enerative **U**nified model for **M**otion **P**laning tasks, namely GUMP, with a generative model structure and a simple tokenizer, which uses an object's unique ID as its key and a compressed state space as its value. This design significantly enhances the model's flexibility, enabling us to adopt an efficient partial AR structure. This model demonstrates high scalability in computational efficiency and shows strong generalization capabilities in understanding complex traffic flows and handling long-tail cases. In addition, its generative ability allows infinite sampling from the learned distribution, which enables long-horizon reactive agent simulation.

Centered at this model, as shown in Fig. 1, we have explored various downstream tasks, and found that this model can be used as:

1. a *data generator* that creates scenarios specific to user's prompts;
2. a *realistic simulator* serving as a reactive closed-loop test bed;
3. a *planner* that unravels interactions between agents to reduce infractions and improve human-like behavior;
4. an *online training* module that enhances the effectiveness of reinforcement learning for policy models.

In summary, our main contributions are threefold. Firstly, we propose a novel generative model that features a simple key-value paired tokenizer. We show that this model achieves state-of-the-art performance on both the simulation and the planning benchmarks. Secondly, we have extensively investigated the usage of this model as a foundation model in a wide range of downstream tasks, and demonstrate that it can significantly improve the functionalities of these tasks. Thirdly, we provide a framework of using the generative model as a central component for developing a closed-loop training and evaluation system. To our best knowledge, we are the first to solve all learning-based motion planning tasks with a unified framework.

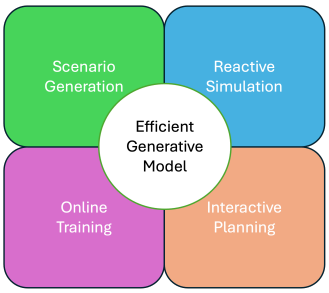

Fig. 1: We are motivated to provide a generative model as the central unit that supports all the learning-based motion planning tasks in the autonomous driving domain. We categorize the tasks into four distinct sub-domains: data generation, model evaluation, model training, and model inference. These sub-domains are visually distinguished in our diagram by different colors—green for data generation, blue for model evaluation, purple for model training, and orange for model inference. Our approach encompasses both offboard applications (the first three sub-domains) and onboard application (the last sub-domain). Specifically, scene generation aims at data generation capable of producing specific traffic scenarios based on context information, such as high-definition maps or user prompts; Reactive simulation aims at a closed-loop evaluator that provides realistic, human-like agents that respond to the behavior of the ego vehicle and its environment; Online training aims at a closed-loop training module that allows the learned policy to interact with environment, collect rewards, and perform back-propagation. Lastly, interactive planning aims at enhancing an onboard planner by parallel unrolling to seek for the optimal trajectory that achieves the highest reward.

2 Methods

2.1 Formulation

Let the dynamic states of the agents at time t be denoted by $s_t = (a_t^{AV}, a_t^{env})$, where a_t^{AV} is the self-driving vehicle, and a_t^{env} is composed of surrounding agents. Specifically, $a_t^{AV} = (a_t^0)$ and $a_t^{env} = (a_t^i) \quad \text{for} \quad i \in \{1, 2, \ldots, n\}$, and the context is denoted as c, which includes a static map [54] and language description

prompts [69]. We factorize the joint probability distribution of traffic scenarios as follows:

$$P(s_t, s_{t-1}, \ldots, s_0, c) = \underbrace{P(s_t|s_{t-1}, \ldots, s_0, c) \cdot \ldots \cdot P(s_1|s_0, c)}_{\text{scene extrapolation}} \cdot \underbrace{P(s_0|c)}_{\text{scene generation}} \cdot \underbrace{P(c)}_{\text{context}}. \quad (1)$$

Naturally, traffic scenarios can be modeled in a probabilistic and sequential manner: given just the context c, we can generate the initial states of the agents s_0; With both the context c and the initial states s_0, we can extrapolate the subsequent states $s_1, s_2, \ldots, s_{t-1}$ of the agents. The initial step allows us to create scenarios through scene generation, while the next step enables us to foresee how these scenarios evolve over time, which we refer to as scene extrapolation.

Scene Generation The initial states can be further factorized as:

$$P(s_0|c) = \prod_j P(a_0^j|a_0^0, \ldots, a_0^{j-1}, c). \quad (2)$$

Specifically, we autoregressively generate the initial state of each agent in the scene, denoted as a_0^i, where $a_0^i = \{x, y, \theta, v_x, v_y, w, l\}$ represents the initial position (x, y), heading (θ), velocity components (v_x, v_y), width (w), and length (l) for the i^{th} agent.

Scene Extrapolation We can further factorize the the scene extrapolation task as [50]:

$$\begin{aligned} P(s_T, s_{T-1}, \ldots|s_0, c) &= P(a_{1:T}^{AV}, a_{1:T}^{envs}|s_0, c) \\ &= \prod_t \pi_{AV}(a_t^{AV}|a_{<t}^{AV}, a_{<t}^{env}, c) \cdot P(a_t^{envs}|a_t^{AV}, a_{<t}^{AV}, a_{<t}^{env}, c). \end{aligned} \quad (3)$$

In this model, π_{AV} represents the policy for autonomous vehicles. Note that this policy can be replaced with any planning policy. Moreover, we can categorize environmental actions a_t^{env} into two distinct groups: $a_t^{tracked}$ and $a_t^{newborn}$. Here, $a_t^{tracked}$ refers to objects that have been previously tracked, while $a_t^{newborn}$ represents objects that have just emerged in the environment.

We enable probabilistic modeling of dynamic scene information without imposing a limit on the number of objects throughout the extrapolation. This capability is crucial for realistically modeling scenarios where objects may disappear or newly appear, addressing both currently obscured objects that may become visible later and those present now but might move out of view. Such a nuanced treatment of dynamic elements significantly improves the ability of our model to handle complex, unpredictable, and long-tail problems in autonomous driving.

Evaluate Planning Policies Suppose we have a series of candidate policies, denoted as π_i, where $i = \{0, 1, \ldots, N_p - 1\}$. These policies can be rule-based or learning-based. Given observations s at a certain time, these policies can produce different response actions $a_i \sim \pi_i(s)$. Under the Markov assumption, the value function $V^{\pi_i}(s)$ of a chosen policy π_i is denoted as:

$$V^{\pi_i}(s) = \sum_{s'} P(s'|s, \pi_i(s)) \left[r(s, \pi_i(s)) + \gamma V^{\pi_i}(s') \right] \tag{4}$$

where γ is the discount factor and $P(s'|s, \pi_i(s))$ is the state transition probability function, which can be modeled with the probabilistic rollouts of our world model. Here, s' represents the next state, and $r(s, \pi_i(s))$ is the reward function, which may be defined as [2]:

$$r(s, \pi_i(s)) = \prod_m \Theta_m(s, \pi_i(s)) \cdot \sum_n \omega_n \Phi_n(s, \pi_i(s)) \tag{5}$$

where Θ is the critical metrics, such as safety and driving directions, and Φ is less critical weighted metrics, such as progress, speed limit, comfort and so on, and ω_n is the metric weight. More details can be found in the Appendix B.2.

Given the vast and probabilistic nature of the state space s, we estimate it by sampling from its overall distribution. As in Eq. 3, the results are referred as rollouts, denoted as $X_i^k = \{s_0, s_1, \ldots, s_T\}_i^k = \{a_t^{AV}, a_t^{env}\}_i^k$, where $i \in \{0, 1, 2, \ldots, N_p - 1\}$, and $k \in \{0, 1, 2, \ldots, N_r - 1\}$, and N_p is the number of the candidate policies, and N_r is the number of the paralleled rollouts for each policy. So, the total number of the rollouts is $N_p \times N_r$. Then we apply the evaluator to calculate the accumulated rewards for each policy following Eq. 4.

Accordingly, the ego vehicle's optimal driving policy can be written as

$$\pi^\star(s_0) = \underset{\pi_i}{\operatorname{argmax}} \hat{V}^{\pi_i}(s_0) \tag{6}$$

where $\hat{V}^{\pi_i}(s_0)$ is the value estimated through rollouts obtained by sampling.

Online Training with Reinforcement Learning In standard Reinforcement Learning (RL), the problem to be solved is typically described as a Markov Decision Process (MDP) [72], which is commonly defined by a tuple $(\mathcal{S}, \mathcal{A}, P, r, \gamma)$, where $\mathcal{S}$, $\mathcal{A}$ denote the state space and action space, respectively. $P(s'|s, a)$ denotes the state transition process, *i.e.*, the probability of transiting to s' given the current state s and action a. r is the reward function and γ is the discount factor. The goal of RL is to learn a policy $\pi(s; \theta)$ that maximizes the accumulated discounted return.

While much effort has been devoted to the development of RL algorithms in the past [23, 64], the importance of high-fidelity simulator starts to gain more attentions recently [53]. For autonomous driving, one challenge in terms of simulation is realistic reactive behaviors of other road participants (*i.e.* the implementation of P), which is arguably hard to be fully characterized by scripted rules [16]. Therefore, instead of implementing P by scripting (*i.e.* $P \triangleq P_{\text{scripted}}$)

as in standard simulator [16], we leverage our data-driven world model for the state transition:

$$P(s'; s, a) \triangleq P_{\text{WorldModel}}(s'; s, a), \tag{7}$$

where $P_{\text{WorldModel}}$ denotes our world model trained on real-world data, therefore capturing the real-world behavior and avoiding cumbersome rule crafting. This could significantly enhance RL by ensuring alignment with real-world logic.

The unroll process could be presented as the following equations:

$$s' \sim P_{\text{WorldModel}}(s'; s, a) \qquad a \sim \pi(s; \theta).$$

And rewards can be calculated with Eq. 4 and Eq. 5. For training, one may use typical RL algorithms such as Soft Actor-Critic (SAC) [23].

2.2 Network

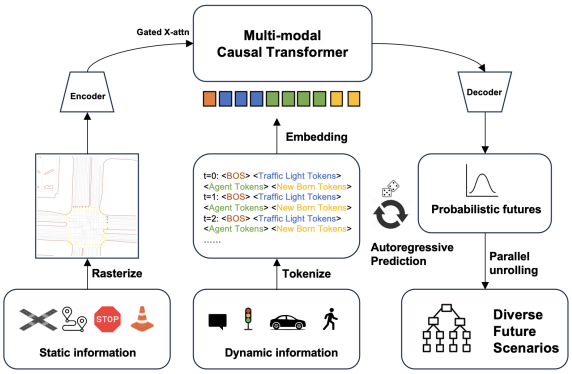

Fig. 2: GUMP is composed of a raster encoder that encodes static information, a key-value pair tokenizer that discretizes dynamic information, a Multimodal Causal Transformer (MCT) that predicts the next token in an autoregressive manner, and a decoder that samples the probabilistic features and decodes to future scenarios.

The overall architecture of our model, depicted in Figure 2, is comprised of four main components: a static raster autoencoder, a dynamic tokenizer, a Multimodal Causal Transformer (MCT), and an auto-regressive decoder. For the static information, elements such as maps, navigation routes, and static obstacles are converted into a raster image. This image is then encoded with a 2D Convolutional Encoder [28]. Meanwhile, dynamic information is encapsulated into a sequence of tokens, providing a linguistic representation of driving behaviors. The static and dynamic information is then fused [1] and encoded with a multimodal visual-language model [60]. Due to its predictive nature, our model can be further accelerated by the intra-frame non-autoregressive (NAR) conversion.

Tokenization In probabilistic modeling of continuous state spaces, discretization is required. Unlike previous studies that use complex transformations to map the state space to the action space and then reintegrate to recover the state space [56, 65], our method directly quantizes the state space.

Choosing state space over action space is motivated by several key factors. Although the action token space is more compact, featuring a smaller vocabulary size, this compactness often sacrifices interpretability. Moreover, encoding with action space requires state-dependent decoding, where each step depends on all preceding actions and the initial states. This dependency may result in compound errors. Additionally, this dependency can also limit the network's flexibility, thereby restricting its generative capabilities, such as predicting traffic lights, handling the emergence and disappearance of elements, or generating scenarios.

In our design, each object is composed of two tokens with distinct functionalities, akin to a "key-value pair": a control token and a state token. The control token serves as the key, used to distinguish between different objects by summing the token embeddings of each object's unique ID and object category. The state token serves as the value, utilized to depict the specific state space of the object. Specifically, we select different state spaces for different types of objects, and perform quantization of the state space for each object. For instance, for traffic lights, the token space includes coordinates (x, y, θ) and traffic signal states (s_{tl}). For traffic participants such as pedestrians and vehicles, the token space includes coordinates (x, y, θ), velocity $(v_x,\ v_y)$, and size $(w,\ l)$. This key-value pair tokenization enables our model with indexing ability. We can selectively query and decode any object of interest or arbitrarily add or remove objects. This approach allows the model to dynamically allocate computational resources for increased efficiency and to manage the disappearance and emergence of objects effectively.

Similar to language models in sequence modeling, we incorporate special tokens, including the beginning of sequence (BOS) token, traffic light end token, and newborn begin token. The final tokenized sequence structure can be depicted as shown in Fig. 2. Additionally, we use a predetermined set of scenario description prompts, which can be represented with a fixed vocabulary. This vocabulary can be embedded similarly to the special tokens.

Token Embedding Each token is embedded in latent features. For the key token of each agent, we sum the embedding features of the agent's unique ID and its class type. For it's value token, we embed all the states with sinusoidal embedding along with tokenized state embeddings. All embeddings are then summed with a learnable positional embedding, as in the equation below:

$$\begin{aligned} e^i_{\text{key}} &= \text{PE}(i) + \sum_{s\in\{\text{id},\text{class}\}} \text{Embed}(s) \\ e^i_{\text{value}} &= \text{PE}(i) + \sum_{s\in\{x,y,\theta,v_x,v_y,w,l\}} (\text{Embed}(s) + \text{sinusoidal}(s)) \end{aligned} \tag{8}$$

Here, i is the position index in the token sequence, PE is the learnable positional embedding, *sinusoidal* is the sinusoidal embedding, and *Embed* is the learnable token embedding. For static objects such as traffic lights, we only embed the coordinates (x, y, θ) and the traffic light status (s_{tl}).

Multi-modal Causal Transformer The Multi-modal Causal Transformer (MCT) module forms the generative core. It builds on the GPT-2 architecture [39, 60] and includes Gated Cross Attention (GCA) Blocks [1] for fusing information from different modalities. We intermittently insert the GCA blocks among self-attention layers to progressively and interactively fuse the dynamic and static information.

RNN Token Decoder To decode the discrete state space S from the compact latent embedding produced by the Multimodal Causal Transformer (MCT), an additional GRU decoder is utilized after the MCT. By querying the MCT with the key embedding for each agent, we extract the corresponding state latent space, which is then autoregressively decoded using stacked GRU layers according to the sequence $\{w, l, v_x, v_y, x_0, y_0, \theta_0, ..., x_T, y_T, \theta_T\}$, where T represents the predicted horizon for each agent. This process facilitates temporal aggregation and intra-frame NAR conversion. To maintain the stability of decoding, we adopted the temperature-scaled top k sampling strategy [18, 29] and dynamic valid masking to ignore invalid tokens.

Prediction Chunking and Temporal Aggregation To combat compounding errors and stabilize the AR process, inspired by the concept of "action chunking" in [84], we have developed a prediction chunking and temporal aggregation module. We perform the chunking process by predicting a longer time horizon with the light-weight RNN described above. During the AR decoding process, we aggregate all the prediction results of each time step as illustrated in Fig. 3. Specifically, we take a weighted average across the different time steps with a decay rate γ; the final output state can be represented as $\hat{s}_t = \sum_{i=0}^{T} \gamma^i \cdot s_{t-i}^i$, where t is the output time step and T is the total time steps of the chunking data. This module significantly improved the model's performance on the Waymo Sim Agents metric [50] as the experimental result shown in Appendix. E.2.

Intra-frame NAR Conversion To speed up a full-AR model, one approach is to decode the parts that are less dependent in parallel [4]. Considering the characteristics of the traffic simulator, we temporarily ignore the current interactions between agents and predict the next state of each agent in parallel with the GRU decoder mentioned earlier. By conditioning on these parallel-decoded states instead, we can eliminate the intra-frame sequential dependency of the AR process, as shown in Fig.4. We can then compensate for the intra-frame dependencies by a single forward pass and solve for more precise agent states

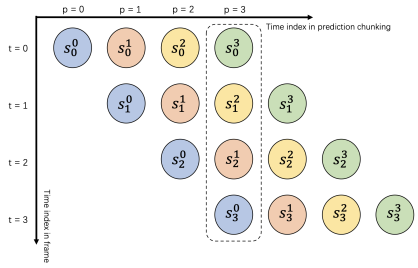

Fig. 3: Prediction Chunking and Temporal Aggregation. s_t^p denotes the per-agent state at time step t and time index p within the chunking data. s_t^p comprises the set $\{x, y, \theta\}$.

efficiently. By converting the full-AR to partial-AR mode, we achieve a significant speedup, as demonstrated in Appendix E.1. Moreover, this conversion is optional, allowing our model to maintain its AR ability, which is essential for scene generation tasks and managing newborn agents. Also, our method avoids altering the causal attention mask, therefore is highly compatible with acceleration libraries [12, 13], leading to notable improvements in training and inference speed and memory usage compared to the versions implemented with customized masking strategy.

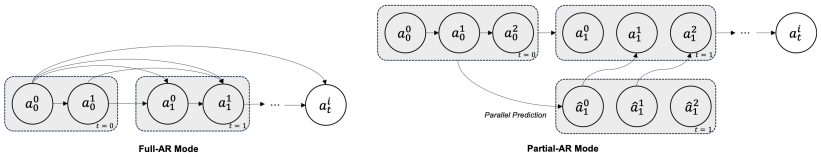

Fig. 4: This figure compares the full-AR mode with the partial-AR mode. Here, a_t^i represents the state of the i^{th} agent at time t. Employing a GRU decoder alongside prediction chunking, we can simultaneously predict the next state of each agent, denoted as $\hat{a}_{t+1}^i$. These predictions serve as surrogate conditions to bypass intra-frame sequential dependencies, markedly speeding up the process by eliminating the need for an intra-frame sequential AR procedure.

2.3 Framework

In this section, we introduce how GUMP serves as a central unit to support a series of downstream tasks, as illustrated in Fig. 5. For those tasks, we have developed specific engines to interact with GUMP, categorized into scene generation, scene extrapolation, planning, and RL engines.

For scene generation, GUMP incorporates a scenario prompt as an additional condition, as described in Sec. 2.2, operating in full-AR mode. Specifically, we se-

quentially construct queried key tokens based on the unique ID and the category of each agent. By querying the MCT, we decode the latent features into initial states. At each iteration, we query and decode information for a single object and use it as the condition for the next query, repeatedly. This AR generation approach effectively manages the relationships between the dynamic objects and the context and can be well controlled by users.

Unlike the task of scene generation, scene extrapolation uses historical scenes as conditions, which can be either from log data or generated by the scene generation task. To accelerate the process, the MCT operates in a partial-AR mode using intra-frame NAR conversion. Through scene extrapolation, we are able to generate diverse, interactive, and closed-loop state predictions for multiple objects within the scene.

This realistic and efficient simulator, denoted as "Plan Engine" in Fig. 5, can be used both off-board as a policy evaluation environment and onboard for deployment, further enhancing existing policies to obtain a more powerful planning policy. This enhancement is achieved by overriding the next state of the ego vehicle in the "Sim Engine" with the action (i.e "a" or "next ego state") output by the existing driving policy, facilitating interaction between the policy and the world model. The resulting rollouts (i.e. "s" or "current states") from this interaction are utilized by the reward module to compute relevant scores. After several rounds of interaction, the model's performance is assessed based on the accumulative score, which can be used for evaluating off-board models or selecting online proposals, as described in Sec 2.1. This highly realistic and probabilistic simulator accurately simulates environmental changes, enabling more precise evaluation and selection of optimal policies.

Unlike "Plan Engine", "RL Engine" employs a trainable ML policy, which undergoes RL training via accumulative rewards. Unlike previous approaches that used IDM [41] or logsim [5], the simulator powered by GUMP offers a more efficient, realistic, diverse, and interactive environment. This reduces the simulation-to-reality domain gap and enhances the model's generalization capability.

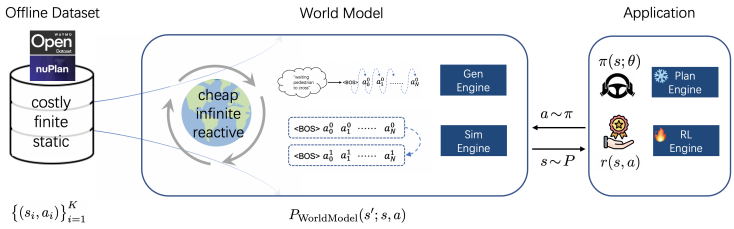

Fig. 5: GUMP serves as a central unit, bridging offline datasets with downstream applications. By learning from the collected offline data, we utilize a generative framework to produce a vast amount of affordable, interactive data, which benefits various downstream tasks.

3 Experiments

Our experiments are primarily conducted on two main public datasets: the Waymo Open Dataset (WOD) [6, 17] and the nuPlan [2]. Our experiments cover scene generation, trajectory prediction, WOD sim agents, and planning. We primarily use the Waymo dataset for the scene generation [19, 71, 73], trajectory prediction and sim agents [50] tasks, and use the nuPlan dataset for prompts conditioned scene generation and planning. You can find more information on these datasets and our benchmarks in the Appendix B, and also related experiment settings and training details in the Appendix C. We have placed more ablation studies, experimental results regards trajectory prediction and qualitative results in Appendix E,G and F, respectively.

Scene Generation As shown in Table 1, we compare our model with other methods on the WOD motion dataset. To ensure a fair comparison, we have closely followed the experimental setting of TrafficGen [19]. Our results significantly outperformed all other competitors across all metrics by a large margin, especially in terms of speed and size, where the error was reduced by 42.1% and 26.6% respectively compared to the previous state-of-the-art performance. This fully demonstrates our model's exceptional capability of creating realistic data.

Method	Position↓	Heading↓	Speed↓	Size↓
SceneGen [73]	0.1362	0.1307	0.1772	0.1190
TrafficGen [19]	0.1192	0.1189	0.1602	0.0932
TrafficGen*	0.1221	0.1174	0.1661	0.0913
GUMP-m	**0.1107**(-9.3%)	**0.099**(-15.7%)	**0.0961**(-42.1%)	**0.0670**(-26.6%)

Table 1: Maximum mean discrepancy (MMD) results on WOD Motion Dataset: Lower MMD Indicate Better Performance Across All Metrics. $\star$: Results reproduced by us under the same experimental setting. GUMP-m refers to the medium variant.

World Simulator We have further validated our model's capability for scene extrapolation as a world simulator on the Waymo Sim Agents Benchmark. As shown in Table 2, our method significantly outperforms competitors in interactive, map-based metrics, and achieves the lowest overall minADE, marking it as state-of-the-art in terms of realism meta metric. The result strongly affirms the exceptional realism and interactivity of our method as a simulator. Moreover, it paves the way for providing a more realistic and interactive environment for various downstream applications.

Interactive Planning To validate the effectiveness of GUMP as a world simulator in interactive planning task, we conducted both open-loop and closed-loop experiments on the large-scale planning dataset, nuPlan. In the open-loop experiments, we treated planning as a trajectory prediction task by unrolling the

Agent Policy	Meta Metrics↑	minADE↓	Kinematic Metrics↑	Interactive Metrics↑	Map-based Metrics↑
Logged Oracle	0.7220	0.000	0.4857	0.8415	0.9043
Constant Velocity	0.2870	7.923	0.0465	0.4087	0.4547
SBTA-ADIA [48]	0.4202	3.611	0.3574	0.4283	0.5087
CAD [11]	0.5314	2.315	0.3357	0.5638	0.7688
Joint-Multipath++ [77]	0.5330	2.049	0.4078	0.5728	0.6677
Wayformer [51]	0.5750	2.498	0.3120	0.7048	0.7747
MTR+++ [59]	0.6077	1.682	0.3597	0.7172	0.8151
MVTA [78]	0.6361	1.869	0.4175	0.7390	0.8139
Trajeglish [56]	**0.6437**	1.615	0.4157	0.7646	0.8104
MVTE⋆ [78]	**0.6448**	1.677	**0.4202**	0.7506	0.8271
GUMP-m	**0.6432**	**1.590**	0.3994	**0.7657**	**0.8290**

Table 2: *Test* set result of WOD Sim Agents Benchmark. Note: ⋆ indicates the use of model ensemble techniques. Methods are ranked by composite metric on the V1 Leaderboard. Our method significantly outperforms others in interactive, map-based metrics, and achieves the lowest overall minADE, marking it as state-of-the-art in terms of realism meta metric. The highest score is highlighted in bold.

predicted ego states. To eliminate randomness, we parallel sampled N times and averaged these rollouts trajectories. As shown in Table 3, compared to other methods, our method achieved the best results in most metrics, demonstrating the feasibility of imitating a planning policy by learning a world simulator.

Methods	Score ↑	8sADE ↓	3sFDE ↓	5sFDE ↓	8sFDE ↓	MR ↓
IDM [75]	37.7	9.600	6.256	10.076	16.993	0.552
PlanCNN [61]	64.0	2.468	0.955	2.486	5.936	0.064
Urban Driver [63]	76.0	2.667	1.497	2.815	5.453	0.064
PDM-Open [14]	85.8	2.375	0.715	2.06	5.296	0.042
CKS-124m [70]	88.0	**1.777**	0.951	2.105	4.515	0.053
CKS-1.5b [70]	86.6	1.783	0.971	2.140	4.460	0.047
GUMP-m	**88.6**	1.820	**0.743**	**1.833**	**4.453**	**0.046**

Table 3: Performance comparison of open-loop planning on the validation 14 set of the nuPlan dataset. Our model outperforms all the other models across most metrics.

We further integrated the imitation policy with a set of rule-based policies, enhanced it with an interactive scoring module powered by GUMP, and validated its effectiveness in a closed-loop reactive environment. We have compared it with previous state-of-the-art results on the test14-hard splits [61], as shown in Table 4. Additionally, we have benchmarked our planner and other competitors in two different reactive environments: IDM and GUMP. The experimental results demonstrate that our policy outperformed all the others in both reactive environments, showing improvements in progress, drivable area compliance, and time to collision, among other metrics, when compared to the previous state-of-the-art PDM [14]. The reliance of PDM on a constant velocity assumption results in overly conservative decision-making. In contrast, our adoption of a stochastic modeling approach allows for better handling of future uncertainties, thereby achieving a better balance between progress and safety (TTC).

Policy	Reactive env	Driving Score	Col.	Driv. Comp.	Dir. Comp.	Making Progress	TTC	Speed Limit	Progress	Comfort
Expert [10]	IDM	68.80	-	-	-	-	-	-	-	-
UrbanDriver [63]	IDM	49.07	-	-	-	-	-	-	-	-
GC-PGP [26]	IDM	39.63	-	-	-	-	-	-	-	-
GameFormer [36]	IDM	68.83	-	-	-	-	-	-	-	-
planTF [10]	IDM	61.70	-	-	-	-	-	-	-	-
hoplan [32]	IDM	75.06	89.33	94.85	96.13	**97.05**	80.51	95.28	85.02	**98.52**
PDM [14]	IDM	75.18	**95.22**	95.58	99.08	93.38	84.19	**99.53**	75.47	83.45
GUMP hybrid	IDM	**77.77**	94.36	**98.98**	98.95	94.41	**87.46**	97.51	**77.08**	79.84
hoplan [32]	GUMP	69.56	93.63	96.25	97.37	86.14	88.01	94.17	**76.74**	**98.87**
PDM [14]	GUMP	72.26	**97.00**	95.50	99.25	88.38	86.89	**99.51**	70.43	85.39
GUMP hybrid	GUMP	**73.60**	94.98	**98.97**	99.31	**88.76**	**89.82**	97.18	72.76	82.00

Table 4: Results on the Hard-14 Test Set: Higher Scores Indicate Better Performance. Our driving score in both the IDM and the GUMP evaluation environments is superior to that of other competitors, and we achieve a better balance between progress and safety (TTC). For detailed metrics definition, please refer to Appendix B.2. In this context, "Col." refers to the collision, "Driv. Comp." refers to the drivable area compliance, and "Dir. Comp." refers to driving direction compliance, and "TTC" stands for time to collision.

Online Training We further demonstrate the effectiveness of using GUMP as an online training environment, as detailed in Table 5. For comparison, we used imitation learning (IL) and the Soft Actor-Critic (SAC) algorithm for online training, all experiments using ResNet18 as the policy model. We selected both logsim [5] and GUMP as our environments for online training and testing, respectively. Comparing exp ID 1, 2 with ID 3, 4, we can conclude that using GUMP as the training environment significantly enhances the RL model's performance across different testing environments, substantially reducing the collision rate and improving progress, and outperforms the IL baseline. In addition, policies trained in the GUMP environment achieved better results in realistic and reactive GUMP testing environments compared to those trained in logsim environments. These improvements are attributed to GUMP's capability to generate a vast amount of realistic, interactive, and diverse training scenarios, greatly enhancing the efficiency of RL training and reducing the sim-to-real domain gap. These findings highlight the effectiveness of GUMP as an environment for online training.

Scaling Laws To validate our model's scalability, we trained three variants of our model with different MCT backbone capacities, i.e. GUMP-small, GUMP-base and GUMP-medium, and measured the cross-entropy loss versus the number of tokens consumed during training, as well as the simulation realism meta metric in the WOD Sim Agents validation set, as illustrated in Fig. 6. We can observe that our model demonstrates high scalability, as both training FLOPs and model capacity increases. Furthermore, an increase in model capacity leads to a significant reduction in training loss and a notable enhancement in the validation meta-metric, especially when contrasting the GUMP-medium with

ID	Policy	Training env	Testing env	Driving Score ↑	Col. Rate↓	Out of Driv. Rate↓	Progress meters↑	Comfort Score↑
0	IL	log data	logsim	0.595	0.248	**0.014**	44.69	0.892
1	SAC	logsim	logsim	0.587	0.259	0.062	49.77	**0.923**
2	SAC	GUMP	logsim	**0.643**	**0.193**	0.077	**56.78**	0.845
3	SAC	logsim	GUMP	0.569	0.246	0.061	44.46	**0.920**
4	SAC	GUMP	GUMP	**0.626**	**0.215**	0.061	**53.58**	0.862

Table 5: In an ablation study on the nuPlan random 1000 splits validation set, we observed significant improvements when using GUMP for online RL training across different environments, compared to non-reactive logsim environments. This enhancement is evident in both logsim and GUMP testing environments. Please note that the metric definition utilized here is more strict than the original nuPlan metrics. For detailed metric definitions, see the Appendix B.2.

the GUMP-base model. This phenomenon suggests that our model still harbors considerable potential for enhancement, as indicated by the scaling law [31,37].

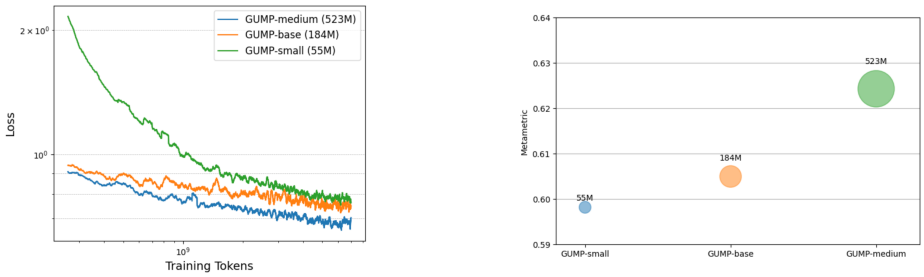

Fig. 6: Left: Training loss versus number of training tokens consumed, with axes shown on a logarithmic scale. Right: WOD Sim Agents meta metric, where a larger area indicates larger model parameters.

4 Conclusion and Future Work

In this work, we have proposed GUMP, a unified generative model designed to solve a broad spectrum of motion planning tasks in the domain of autonomous driving. Our model has demonstrated exceptional scalability and effectiveness across various downstream applications, showcasing its potential to serve as a foundation model in this field.

Limitations and Future Directions: Firstly, to enhance the model's efficiency through engineering efforts such as model quantization and key-value caching. Additionally, adopting vectorized map inputs instead of raster map inputs may potentially yield more accurate map information. Furthermore, directly integrating sensor data enables truly end-to-end modeling and potentially enhances its applicability and performance in real-world settings.

References

1. Alayrac, J.B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al.: Flamingo: a visual language model for few-shot learning. Advances in Neural Information Processing Systems **35**, 23716–23736 (2022) 6, 8
2. Caesar, H., Kabzan, J., Tan, K., et al.: Nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles. In: CVPR ADP3 workshop (2021) 5, 11, 21, 23, 25
3. Chai, Y., Sapp, B., Bansal, M., Anguelov, D.: Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. In: Conference on Robot Learning. pp. 86–99. PMLR (2020) 31
4. Chang, H., Zhang, H., Jiang, L., Liu, C., Freeman, W.T.: Maskgit: Masked generative image transformer. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2022) 8
5. Chen, D., Koltun, V., Krähenbühl, P.: Learning to drive from a world on rails. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15590–15599 (2021) 10, 13, 22
6. Chen, K., Ge, R., Qiu, H., Ai-Rfou, R., Qi, C.R., Zhou, X., Yang, Z., Ettinger, S., Sun, P., Leng, Z., Mustafa, M., Bogun, I., Wang, W., Tan, M., Anguelov, D.: Womd-lidar: Raw sensor dataset benchmark for motion forecasting. arXiv preprint arXiv:2304.03834 (April 2023) 11
7. Chen, Y., Karkus, P., Ivanovic, B., Weng, X., Pavone, M.: Tree-structured policy planning with learned behavior models (2023) 22
8. Chen, Y., Rosolia, U., Ubellacker, W., Csomay-Shanklin, N., Ames, A.D.: Interactive multi-modal motion planning with branch model predictive control. IEEE Robotics and Automation Letters **7**(2), 5365–5372 (2022) 22
9. Chen, Z., Ye, M., Xu, S., Cao, T., Chen, Q.: Deepemplanner: An end-to-end em motion planner with iterative interactions. arXiv e-prints pp. arXiv–2311 (2023) 22
10. Cheng, J., Chen, Y., Mei, X., Yang, B., Li, B., Liu, M.: Rethinking imitation-based planners for autonomous driving (2023) 13
11. Chiu, H., Smith, S.F.: Collision avoidance detour: A solution for 2023 waymo open dataset challenge - sim agents. Tech. rep., Carnegie Mellon University (2023) 12, 39
12. Dao, T.: FlashAttention-2: Faster attention with better parallelism and work partitioning (2023) 9
13. Dao, T., Fu, D.Y., Ermon, S., Rudra, A., Ré, C.: FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In: Advances in Neural Information Processing Systems (2022) 9
14. Dauner, D., Hallgarten, M., Geiger, A., Chitta, K.: Parting with misconceptions about learning-based vehicle motion planning. In: Conference on Robot Learning (CoRL) (2023) 12, 13, 22, 29
15. Devaranjan, J., Kar, A., Fidler, S.: Meta-sim2: Unsupervised learning of scene structure for synthetic data generation. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16. pp. 715–733. Springer (2020) 21
16. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: Carla: An open urban driving simulator. In: Conference on robot learning. pp. 1–16. PMLR (2017) 5, 6, 21

17. Ettinger, S., Cheng, S., Caine, B., Liu, C., Zhao, H., Pradhan, S., Chai, Y., Sapp, B., Qi, C.R., Zhou, Y., Yang, Z., Chouard, A., Sun, P., Ngiam, J., Vasudevan, V., McCauley, A., Shlens, J., Anguelov, D.: Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 9710–9719 (October 2021) 11
18. Fan, A., Lewis, M., Dauphin, Y.: Hierarchical neural story generation. arXiv preprint arXiv:1805.04833 (2018) 8
19. Feng, L., Li, Q., Peng, Z., Tan, S., Zhou, B.: Trafficgen: Learning to generate diverse and realistic traffic scenarios. In: 2023 IEEE International Conference on Robotics and Automation (ICRA). pp. 3567–3575. IEEE (2023) 11, 21, 22, 23, 28
20. Gao, Z., Mu, Y., Shen, R., Chen, C., Ren, Y., Chen, J., Li, S.E., Luo, P., Lu, Y.: Enhance sample efficiency and robustness of end-to-end urban autonomous driving via semantic masked world model. arXiv preprint arXiv:2210.04017 (2022) 22
21. Gu, J., Sun, C., Zhao, H.: Densetnt: End-to-end trajectory prediction from dense goal sets. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15303–15312 (2021) 2, 21
22. Ha, D., Schmidhuber, J.: Recurrent world models facilitate policy evolution. In: Advances in Neural Information Processing Systems, pp. 2451–2463. Curran Associates, Inc. (2018), https://papers.nips.cc/paper/7512-recurrent-world-models-facilitate-policy-evolution, https://worldmodels.github.io 21
23. Haarnoja, T., Zhou, A., Abbeel, P., Levine, S.: Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. arXiv preprint arXiv:1801.01290 (2018) 5, 6, 22, 29
24. Hafner, D., Lillicrap, T.P., Norouzi, M., Ba, J.: Mastering atari with discrete world models. In: International Conference on Learning Representations (2020) 21, 22
25. Hafner, D., Pasukonis, J., Ba, J., Lillicrap, T.: Mastering diverse domains through world models. arXiv preprint arXiv:2301.04104 (2023) 22
26. Hallgarten, M., Stoll, M., Zell, A.: From prediction to planning with goal conditioned lane graph traversals. arXiv preprint arXiv:2302.07753 (2023) 13
27. Hawke, J., Badrinarayanan, V., Kendall, A., et al.: Reimagining an autonomous vehicle. arXiv preprint arXiv:2108.05805 (2021) 1
28. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016) 6
29. Hinton, G.E., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. ArXiv **abs/1503.02531** (2015), https://api.semanticscholar.org/CorpusID:7200347 8
30. Hu, A., Corrado, G., Griffiths, N., Murez, Z., Gurau, C., Yeo, H., Kendall, A., Cipolla, R., Shotton, J.: Model-based imitation learning for urban driving. In: Advances in Neural Information Processing Systems (NeurIPS) (2022) 22
31. Hu, A., Russell, L., Yeo, H., Murez, Z., Fedoseev, G., Kendall, A., Shotton, J., Corrado, G.: Gaia-1: A generative world model for autonomous driving. arXiv preprint arXiv:2309.17080 (2023) 14, 21
32. Hu, Y., Li, K., Liang, P., Qian, J., Yang, Z., Zhang, H., Shao, W., Ding, Z., Xu, W., Liu, Q.: Imitation with spatial-temporal heatmap: 2nd place solution for nuplan challenge. arXiv preprint arXiv:2306.15700 (2023) 13
33. Hu, Y., Shao, W., Jiang, B., Chen, J., Chai, S., Yang, Z., Qian, J., Zhou, H., Liu, Q.: Hope: Hierarchical spatial-temporal network for occupancy flow prediction. arXiv preprint arXiv:2206.10118 (2022) 21

34. Hu, Y., Yang, J., Chen, L., Li, K., Sima, C., Zhu, X., Chai, S., Du, S., Lin, T., Wang, W., Lu, L., Jia, X., Liu, Q., Dai, J., Qiao, Y., Li, H.: Planning-oriented autonomous driving. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2023) 2, 21, 22
35. Huang, Z., Karkus, P., Ivanovic, B., Chen, Y., Pavone, M., Lv, C.: Dtpp: Differentiable joint conditional prediction and cost evaluation for tree policy planning in autonomous driving. arXiv preprint arXiv:2310.05885 (2023) 22
36. Huang, Z., Liu, H., Lv, C.: Gameformer: Game-theoretic modeling and learning of transformer-based interactive prediction and planning for autonomous driving. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 3903–3913 (October 2023) 13, 22
37. Kaplan, J., McCandlish, S., Henighan, T., Brown, T.B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., Amodei, D.: Scaling laws for neural language models. arXiv preprint arXiv:2001.08361 (2020) 14
38. Kar, A., Prakash, A., Liu, M.Y., Cameracci, E., Yuan, J., Rusiniak, M., Acuna, D., Torralba, A., Fidler, S.: Meta-sim: Learning to generate synthetic datasets. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4551–4560 (2019) 21
39. Karpathy, A.: nanogpt. https://github.com/karpathy/nanoGPT (2023) 8
40. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014) 30
41. Li, Q., Peng, Z., Feng, L., Zhang, Q., Xue, Z., Zhou, B.: Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning. IEEE Transactions on Pattern Analysis and Machine Intelligence (2022) 10, 21, 22
42. Li, X., Zhang, Y., Ye, X.: Drivingdiffusion: Layout-guided multi-view driving scene video generation with latent diffusion model. arXiv preprint arXiv:2310.07771 (2023) 2, 21
43. Liao, B., Chen, S., Wang, X., Cheng, T., Zhang, Q., Liu, W., Huang, C.: Maptr: Structured modeling and learning for online vectorized hd map construction. In: International Conference on Learning Representations (2023) 2
44. Liao, B., Chen, S., Zhang, Y., Jiang, B., Zhang, Q., Liu, W., Huang, C., Wang, X.: Maptrv2: An end-to-end framework for online vectorized hd map construction. arXiv preprint arXiv:2308.05736 (2023) 2
45. Liu, Y., Yuantian, Y., Wang, Y., Wang, Y., Zhao, H.: Vectormapnet: End-to-end vectorized hd map learning. In: International conference on machine learning. PMLR (2023) 2
46. Lopez, P.A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., Wießner, E.: Microscopic traffic simulation using sumo. In: 2018 21st international conference on intelligent transportation systems (ITSC). pp. 2575–2582. IEEE (2018) 21
47. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (2018) 33
48. Mo, X., Liu, H., Huang, Z., Lv, C.: Simulating behaviors of traffic agents for autonomous driving via interactive autoregression. Tech. rep., Nanyang Technological University, (2023) 12, 39
49. Mobileye: Mobileye under the hood. https://www.mobileye.com/ces-2024/ (2024) 1
50. Montali, N., Lambert, J., Mougin, P., Kuefler, A., Rhinehart, N., Li, M., Gulino, C., Emrich, T., Yang, Z., Whiteson, S., et al.: The waymo open sim agents challenge. arXiv preprint arXiv:2305.12032 (2023) 2, 4, 8, 11, 21, 24

51. Nayakanti, N., Al-Rfou, R., Zhou, A., Goel, K., Refaat, K.S., Sapp, B.: Wayformer: Motion forecasting via simple & efficient attention networks. In: ICRA (2023) 2, 12, 21, 39
52. Ngiam, J., Caine, B., Vasudevan, V., Zhang, Z., Chiang, H.T.L., Ling, J., Roelofs, R., Bewley, A., Liu, C., Venugopal, A., et al.: Scene transformer: A unified architecture for predicting multiple agent trajectories. ICLR (2021) 2, 21, 38
53. Niu, H., Sharma, S., Qiu, Y., Li, M., Zhou, G., HU, J., Zhan, X.: When to trust your simulator: Dynamics-aware hybrid offline-and-online reinforcement learning. In: Oh, A.H., Agarwal, A., Belgrave, D., Cho, K. (eds.) Advances in Neural Information Processing Systems (2022) 5
54. OpenStreetMap contributors: Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org (2017) 2, 3
55. Pan, M., Zhu, X., Wang, Y., Yang, X.: Iso-dream: Isolating and leveraging noncontrollable visual dynamics in world models. Advances in neural information processing systems **35**, 23178–23191 (2022) 22
56. Philion, J., Peng, X.B., Fidler, S.: Trajeglish: Learning the language of driving scenarios. arXiv preprint arXiv:2312.04535 (2023) 2, 7, 12, 21, 39
57. Prakash, A., Boochoon, S., Brophy, M., Acuna, D., Cameracci, E., State, G., Shapira, O., Birchfield, S.: Structured domain randomization: Bridging the reality gap by context-aware synthetic data. In: 2019 International Conference on Robotics and Automation (ICRA). pp. 7249–7255. IEEE (2019) 21
58. Pronovost, E., Ganesina, M.R., Hendy, N., Wang, Z., Morales, A., Wang, K., Roy, N.: Scenario diffusion: Controllable driving scenario generation with diffusion. Advances in Neural Information Processing Systems **36** (2024) 21
59. Qian, C., Xiu, D., Tian, M.: A simple yet effective method for simulating realistic multi-agent behaviors. Tech. rep. (2023) 12, 21, 39
60. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners (2019) 2, 6, 8
61. Renz, K., Chitta, K., Mercea, O.B., Koepke, A., Akata, Z., Geiger, A.: Plant: Explainable planning transformers via object-level representations. In: 6th Annual Conference on Robot Learning. pp. 459–470. MLResearchPress (2022) 12
62. Santana, E., Hotz, G.: Learning a driving simulator. arXiv preprint arXiv:1608.01230 (2016) 21
63. Scheel, O., Bergamini, L., Wolczyk, M., Osiński, B., Ondruska, P.: Urban driver: Learning to drive from real-world demonstrations using policy gradients. In: Conference on Robot Learning. pp. 718–728. PMLR (2022) 12, 13
64. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017) 5, 22
65. Seff, A., Cera, B., Chen, D., Ng, M., Zhou, A., Nayakanti, N., Refaat, K.S., Al-Rfou, R., Sapp, B.: Motionlm: Multi-agent motion forecasting as language modeling. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8579–8590 (2023) 2, 7, 21
66. Shao, H., Wang, L., Chen, R., Waslander, S.L., Li, H., Liu, Y.: Reasonnet: End-to-end driving with temporal and global reasoning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13723–13733 (2023) 22
67. Shi, S., Jiang, L., Dai, D., Schiele, B.: Motion transformer with global intention localization and local movement refinement. Advances in Neural Information Processing Systems (2022) 2, 21, 22, 38

68. Shi, S., Jiang, L., Dai, D., Schiele, B.: Mtr++: Multi-agent motion prediction with symmetric scene modeling and guided intention querying. IEEE Transactions on Pattern Analysis and Machine Intelligence (2024) 21
69. Sima, C., Renz, K., Chitta, K., Chen, L., Zhang, H., Xie, C., Luo, P., Geiger, A., Li, H.: Drivelm: Driving with graph visual question answering. arXiv preprint arXiv:2312.14150 (2023) 4
70. Sun, Q., Zhang, S., Ma, D., Shi, J., Li, D., Luo, S., Wang, Y., Xu, N., Cao, G., Zhao, H.: Large trajectory models are scalable motion predictors and planners. arXiv preprint arXiv:2310.19620 (2023) 12, 38
71. Suo, S., Regalado, S., Casas, S., Urtasun, R.: Trafficsim: Learning to simulate realistic multi-agent behaviors. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10395–10404. IEEE Computer Society (2021) 2, 11, 21
72. Sutton, R.S., Barto, A.G.: Reinforcement learning: An introduction. MIT press (2018) 5
73. Tan, S., Wong, K., Wang, S., Manivasagam, S., Ren, M., Urtasun, R.: Scenegen: Learning to generate realistic traffic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 892–901 (2021) 11, 21
74. Tesla: Tesla AI Day. https://www.youtube.com/watch?v=ODSJsviD_SU (2022) 1, 22
75. Treiber, M., Hennecke, A., Helbing, D.: Congested traffic states in empirical observations and microscopic simulations. Physical review E **62**(2), 1805 (2000) 12, 22
76. Varadarajan, B., Hefny, A., Srivastava, A., Refaat, K.S., Nayakanti, N., Cornman, A., Chen, K., Douillard, B., Lam, C.P., Anguelov, D., et al.: Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction. In: 2022 International Conference on Robotics and Automation (ICRA). pp. 7814–7821. IEEE (2022) 21
77. Wang, W., Zhen, H.: Joint-multipath++ for simulation agents. Tech. rep. (2023) 2, 12, 39
78. Wang, Y., Zhao, T., Yi, F.: Multiverse transformer: 1st place solution for waymo open sim agents challenge 2023. arXiv preprint arXiv:2306.11868 (2023) 12, 21, 39
79. Xpeng: Xpeng CVPR2023 Workshops. https://www.youtube.com/watch?v=d6ucRgDDUWQ&list=PL3N9otbGBVLc3jdm6yrPtCWdE7C8AsNAy&index=6 (2023) 1
80. Xu, W., Yu, H., Zhang, H., Hong, Y., Yang, B., Zhao, L., Bai, J., ALF-contributors: ALF: Agent Learning Framework (2021), https://github.com/HorizonRobotics/alf 29
81. Yan, W., Zhang, Y., Abbeel, P., Srinivas, A.: Videogpt: Video generation using vq-vae and transformers. arXiv preprint arXiv:2104.10157 (2021) 21
82. Yang, Q.I., Koutsopoulos, H.N.: A microscopic traffic simulator for evaluation of dynamic traffic management systems. Transportation Research Part C: Emerging Technologies **4**(3), 113–129 (1996) 21
83. Zhang, L., Xiong, Y., Yang, Z., Casas, S., Hu, R., Urtasun, R.: Learning unsupervised world models for autonomous driving via discrete diffusion. arXiv preprint arXiv:2311.01017 (2023) 21
84. Zhao, T.Z., Kumar, V., Levine, S., Finn, C.: Learning fine-grained bimanual manipulation with low-cost hardware. arXiv preprint arXiv:2304.13705 (2023) 8