Text-to-Drive: Diverse Driving Behavior Synthesis via Large Language Models

Anonymous CVPR submission

Paper ID *****

Abstract

Adopting language descriptions to generate driving behaviors can offer a scalable and intuitive method for human operators to simulate varied driving scenarios. However, the scarcity of large-scale annotated language-trajectory data makes this approach challenging. To address this gap, we propose Text-to-Drive (T2D) to synthesize diverse driving behaviors via Large Language Models (LLMs). We introduce a knowledge-driven approach that operates in two stages. First, we employ the embedded knowledge of LLMs to generate diverse descriptions of driving behaviors. Then, we leverage LLM's reasoning capabilities to synthesize them into simulation. At its core, T2D employs an LLM to construct a state chart that maps low-level states to high-level abstractions. This strategy aids in downstream tasks such as summarizing low-level observations, assessing policy alignment with behavior description, and shaping the auxiliary reward, all without needing human supervision. With our knowledge-driven approach, we demonstrate that T2D generates more diverse trajectories compared to other baselines and offers a natural language interface that allows for incorporating human preference. Please check our website for more examples: here

1. Introduction

In this work, we utilize the embedded knowledge of Large Language Models (LLMs) to generate diverse descriptions of driving behaviors and then synthesize them in simulation.

Simulators. Simulators have emerged as an effective tool for training and evaluating safety-critical systems, such as autonomous vehicles. Model-based simulators [8, 22] are capable of modeling real-world physics and constructing photorealistic environments. Continual improvements have integrated tools like Scenic [12]. However, these simulators often struggle with a sim-to-real domain gap [31]. In response, data-driven simulators [1, 2, 13, 15, 20] can bridge this gap by leveraging real-world driving data [4, 17, 27] to reconstruct real-world scenes and synthesize novel views.

Behavior Generation. Despite their advantages, current data-driven simulators are unable to control the behaviors of surrounding vehicles. Recent advancements in data-driven traffic generation methods have employed neural networks to synthesize new scenarios [3, 10, 23, 28, 29, 32], offering avenues for more realistic traffic modeling. Complementing these efforts, research has expanded into multiple directions: learning latent representations from driving datasets [7, 16], incorporating human preferences with reinforcement learning [5], and exploring generative scenarios through diffusion models [5, 6]. These efforts collectively enhance the controllability of simulated environments, yet they cannot take text descriptions of driving behaviors as inputs. The necessity of manual labeling for driving behaviors in recent research [7, 25] further highlights this limitation.

More recently, language-conditioned traffic generation has been explored in [24, 30, 34]. [30] leverages LLMs to translate textual descriptions of traffic scenes directly into driving trajectories. However, while their approaches focus on low-level trajectory generation, ours explores the generation of diverse high-level behaviors such as "tailgating". In this work, we build upon the capabilities of LLMs for zero-shot generation of reward functions [19, 26, 33]. Our research explores this capability further and extends it to diverse driving behaviors for simulation, especially in scenarios lacking a ground-truth fitness function.

We introduce ***T2D***, a knowledge-driven approach that utilizes LLMs to generate diverse descriptions of driving behaviors and then synthesizes them in simulation. This approach complements data-driven simulators that depend only on human-driving data. Given a behavior description, ***T2D*** generates a mapping of low-level states (e.g: vehicle position, heading, speed) to high-level abstractions (e.g. "on the on-ramp", "near the end of on-ramp", and "merged"). By leveraging this abstract state representation, transitions are defined to capture the temporal dynamics of the behavior, effectively embodying temporal logic. Using ***T2D***, we generated 18 behaviors from language descriptions. We make the following contributions:

- We introduce ***T2D***, a knowledge-driven method for sim-

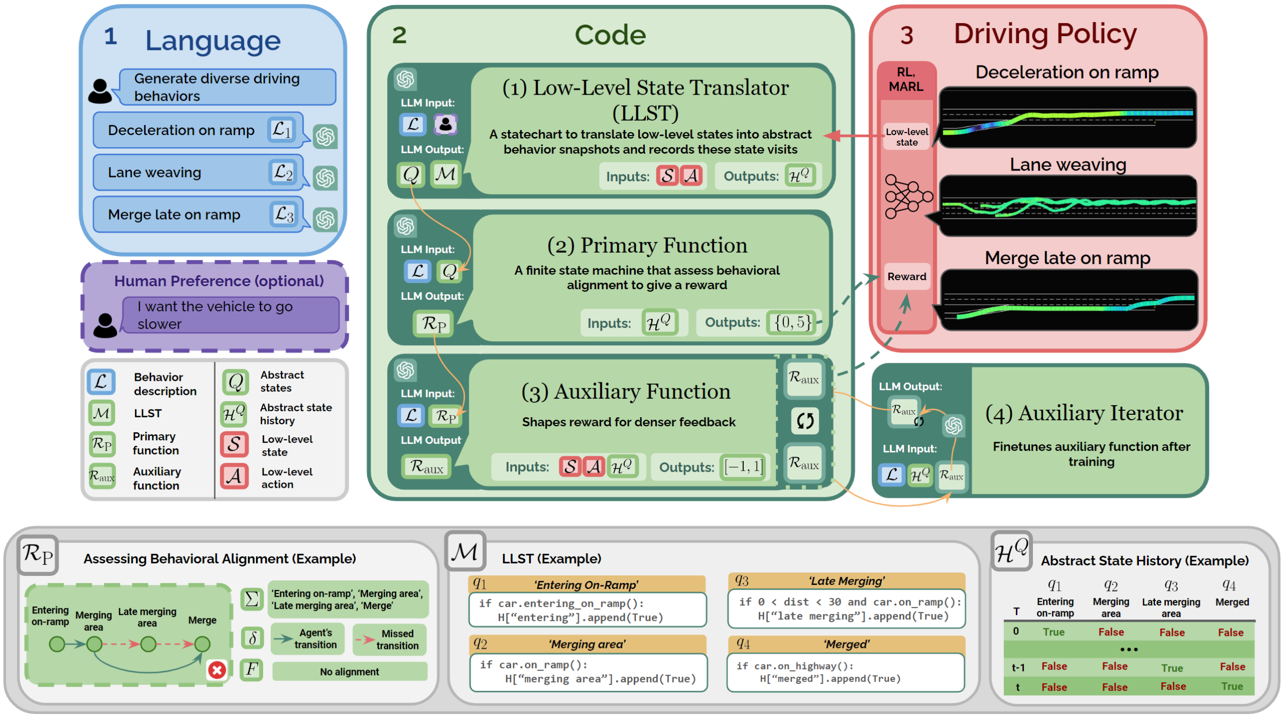


Figure 1. **Overview.** *Left*: First, an LLM generates diverse descriptions of driving behaviors, which can incorporate human preferences through a natural language interface. *Middle*: Next, an LLM generates a low-level state translator (LLST), primary function, and auxiliary function from a description of a driving behavior. The LLST translates low-level states to abstract states (see example in *bottom middle* block) and then records their state visit history (see example in *bottom right* block). The primary function gives a reward only when the vehicle exhibits the target behavior, using a finite-state machine for formal verification of behavior emergence (see example in *bottom left* block). The auxiliary function provides rewards for reaching intermediate states and can be iteratively updated. *Right*: Finally, we employ a standard multi-agent RL framework to train a driving policy using the primary and auxiliary functions as guidance.

ulation that enables (i) text-to-driving behavior synthesis, and (ii) diverse driving behavior generation.
- Our method facilitates the use of LLM-based reasoning by encapsulating the logic in state machines. This facilitates complex policy training processes such as: (i) summarizing low-level observations, (ii) reasoning about behavioral alignment, and (iii) iteratively updating the auxiliary function, without any human supervision.
- We demonstrate our method effectively retains the behavioral context across natural language, code, and driving policy, enabling it to simulate a driving behavior from a description. Additionally, ***T2D*** not only generates more diverse trajectories compared to baselines but also offers a language interface to integrate human preferences.

2. METHOD

Generating Behavior Descriptions. In our knowledge-driven approach, we use `gpt-4` to generate descriptions of diverse driving behaviors, $\mathcal{L} \sim \pi^L(.|\text{scene})$ from a description of a scene.

Low-Level State Translator. Given a behavior description $\mathcal{L}$, our code generation model π^C, generates a low-level state translator, $\mathcal{M} \sim \pi^C(.|\mathcal{L})$, which has three responsibilities: First, $\mathcal{M}$ decomposes the behavior into abstract states, Q. Each abstract state, $q \in Q$, captures an essential aspect of the driving behavior. For example, in the case of "merging late on the on-ramp", q could be any of "on the on-ramp," "merging," and "near the end of the on-ramp" as illustrated in Figure 1. Second, $\mathcal{M}$ is constructed by the LLM as a statechart, mapping lower-level states to abstract states. This statechart is defined as a tuple $\mathcal{M} = (Q, \mathcal{T}, E, U, G)$, where $\mathcal{T}$ is the set of transitions triggered by a low-level event E, conditioned on a guard in G which are boolean functions that return true under certain conditions Finally, the update action in U records each abstract state visit as a boolean value over a rollout of T timesteps in the state history, $\mathcal{H}^Q : Q \rightarrow \{\text{true}, \text{false}\}^T$. This is extremely effective at summarizing low-level observations back to the code and language space (see Figure 2).

Primary Reward Function. The primary function $\mathcal{R}_\text{P}$, generated by an LLM, takes $\mathcal{H}^Q$ as input and returns a reward, $\mathcal{R}_\text{P} : \mathcal{H}^Q \rightarrow \{0, 5\}$. This function serves two purposes: first, it assesses the behavioral alignment of the driving policy π^P with the target behavior $\mathcal{L}$; second, it awards a large reward when the vehicle demonstrates the target behavior to guide the driving policy. To generate

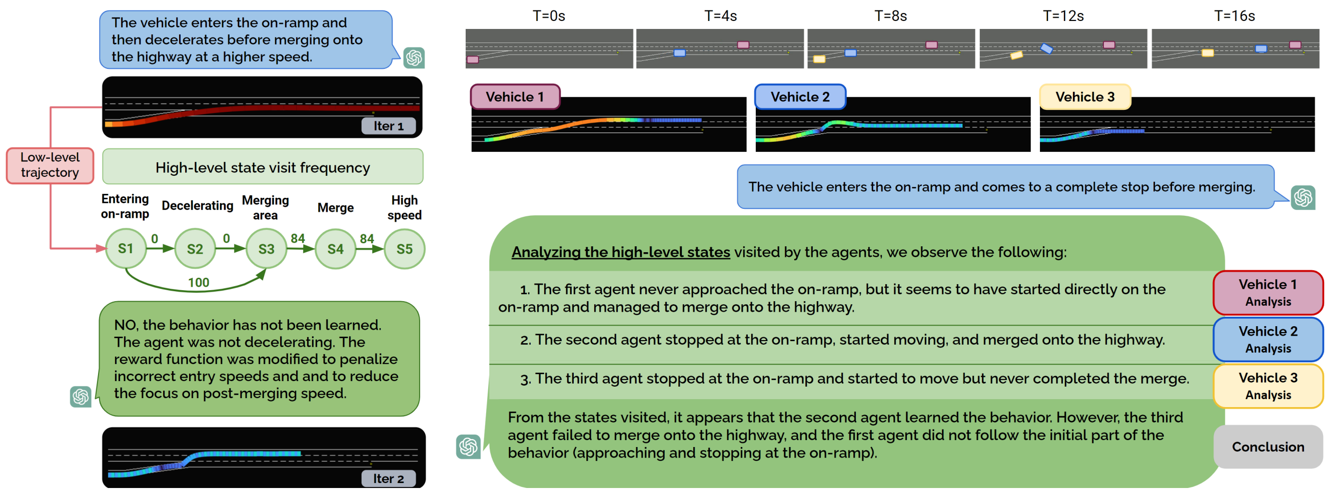
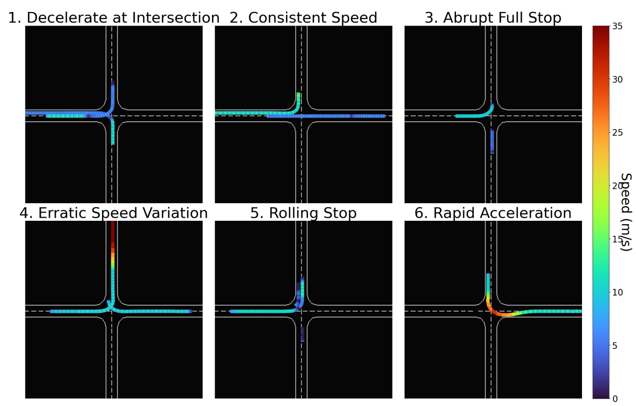


Figure 2. *Left*: The auxiliary iterator LLM analyzes the policy after training to decide whether and how to adjust the auxiliary function based on the history of abstract state visits. *Right*: The right figure illustrates the LLM's reasoning process, where it reads a high-level behavior sequence, analyzes it, and then provides an accurate summary of the low-level trajectories.



Behaviors	Emergence Rate (%)	Collision Rate (%)	Avg. Speed (m/s)
1. Decelerate through intersection	56.67	40.00	7.42
2. Consistent speed crossing	63.33	20.00	5.73
3. Abrupt full stop at intersection	56.67	13.33	1.06
4. Erratic speed	93.33	26.67	5.60
5. Rolling stop at intersection	73.33	30.00	5.10
6. Rapid acceleration at intersection	76.67	43.33	21.24

Figure 3. Diverse driving behaviors at an intersection.

$\mathcal{R}_{\text{P}} \sim \pi^C(.|(Q, \mathcal{L}))$, we combine the abstract state names Q, and behavior description $\mathcal{L}$, as inputs to π^C. The LLM constructs $\mathcal{R}_{\text{P}}$ as a finite-state machine (FSM) that models the target behavior $\mathcal{L}$. This FSM can be described as a tuple, $\mathcal{R}_{\text{P}} = (Q, \Sigma, \delta, q_0, F)$; where $\Sigma = \{q | q \in Q\}$ is the input alphabet, $\delta : Q \times \Sigma \rightarrow Q$ is the transition function. Upon reaching the accepting states, F, the FSM awards a reward of 5 to the vehicle. The formal structure of the FSM provides a framework for verifying the abstract behavior sequences given by $\mathcal{H}^Q$.

Auxiliary Reward Function. To improve exploration efficiency, we use an auxiliary function, $\mathcal{R}_{\text{aux}} : \mathcal{S}, \mathcal{A}, \mathcal{H}^Q \rightarrow [-1, 1]$, that takes low-level state $\mathcal{S}$, action $\mathcal{A}$, and history $\mathcal{H}^Q$, as inputs and returns a reward. Our auxiliary function is generated using π^C, with $\mathcal{R}_{\text{P}}$ and $\mathcal{L}$ as inputs, $\mathcal{R}_{\text{aux}} \sim \pi^C(.|(\mathcal{R}_{\text{P}}, \mathcal{L}))$.

Auxiliary Function Iterator. To mitigate unintended behaviors from reward shaping, we iteratively update $\mathcal{R}_{\text{aux}}$ using a code generation model, $\mathcal{R}'_{\text{aux}} \sim \pi^C(.|(\mathcal{R}_{\text{aux}}, \mathcal{H}^q, \mathcal{L}))$; where $\mathcal{R}'_{\text{aux}}$ is the new auxiliary function, and $(\mathcal{R}_{\text{aux}}, \mathcal{H}^q, \mathcal{L})$ are the inputs to the LLM. The history $\mathcal{H}^q$ provides the LLM with high-level insights into a rollout, allowing it to adjust the reward incentive structure accordingly. For an illustration, see the left of Figure 2.

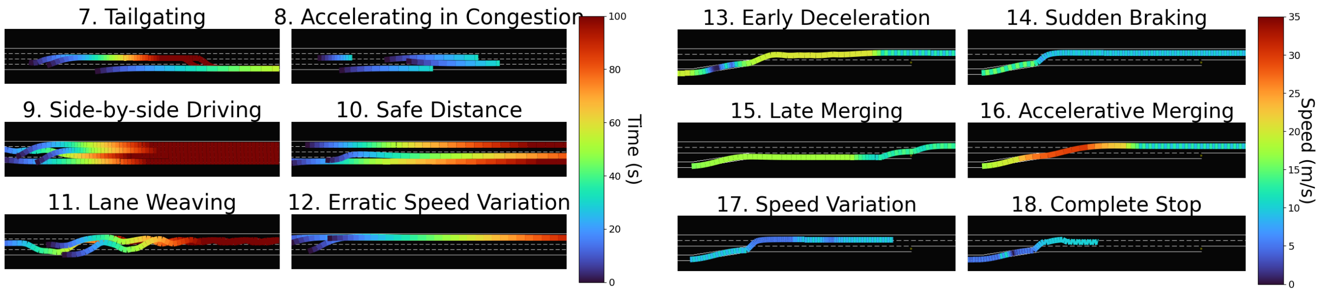
Training Driving Policy. We employed a multi-agent implementation of the Advantage Actor-Critic algorithm (MAA2C) [21] to learn the driving policies π^P. We employ a concurrent training strategy in a cooperative environment under partial observation conditions.

3. EXPERIMENTS

Policy Alignment. In Figure 5a, we show that there is strong agreement between the language and code domain, we compute the pairwise cosine similarity between the sentence embedding of $\mathcal{L}$ and the code embedding of $\mathcal{R}_{\text{aux}}$ using CodeBERT [11]. In addition, Figure 5b shows a strong agreement between code and driving policy domain. We quantify agreement by computing the expected cumulative reward of π^P using $\mathcal{R}_{\text{aux}}$. Specifically, we define the element $\mathbf{A}_{ij}^{C \leftrightarrow P}$ of the agreement matrix $\mathbf{A}^{C \leftrightarrow P}$ as:

$$\mathbf{A}_{ij}^{C \leftrightarrow P} = \mathbb{E}_{\tau \sim \pi_j^P} \left[\sum_{t=1}^{T} \mathcal{R}_i(\mathcal{S}_t, \mathcal{A}_t) \right] \quad (1)$$

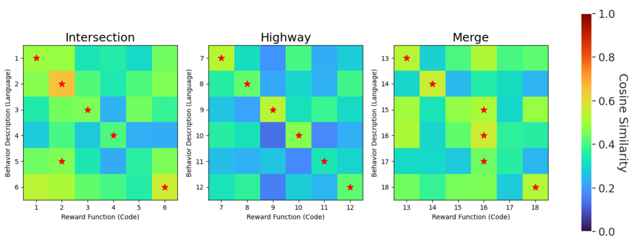
Then, we compare these values relative to the performance of alternate driving policies on the same reward function.



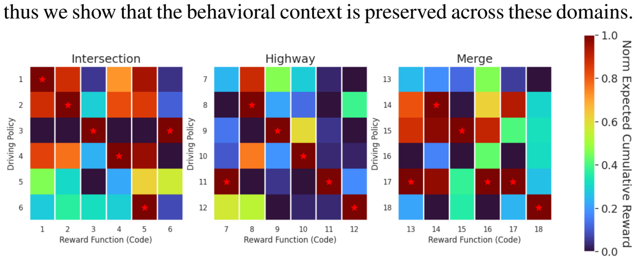
Behaviors	Emergence Rate (%)	Collision Rate (%)	Avg. Speed (m/s)
Tailgating	40.00	56.67	19.62
Accelerating in Congestion	100.00	73.33	30.18
Side-by-Side Driving	63.33	33.33	14.28
Following at a Safe Distance	93.33	0.00	31.79
Lane Weaving	73.33	36.67	15.52
Erratic Speed	53.00	63.33	29.96

Behaviors	Emergence Rate (%)	Collision Rate (%)	Avg. Speed (m/s)
Early Deceleration on Ramp	80.00	30.00	12.46
Sudden Braking After Merging	80.00	13.33	9.79
Late Merging at Ramp End	86.67	0.00	13.17
Accelerative Merging	60.00	33.33	12.43
Merging with Speed Variation	96.67	6.67	8.72
Merging from Complete Stop	60.00	26.67	9.05

Figure 4. Diverse highway driving and merging behaviors.



(a) A diagonal line in the code and language agreement matrix indicates that there is a high similarity between the text description and code, and thus we show that the behavioral context is preserved across these domains.



(b) A diagonal line in the code and driving policy agreement matrix indicates that the policy trained by the reward function was most optimal compared to the other evaluated policies, and therefore we show that the behavioral context is preserved across these domains.

Figure 5. Agreement matrix to show behavioral alignment.

Formally, we consider a reward function $\mathcal{R}_i$ to be in agreement with a policy π^P_j if $\forall k \neq j, \mathbf{A}^{C\leftrightarrow P}_{ij} > \mathbf{A}^{C\leftrightarrow P}_{ik}$. The intuition is to examine how each driving policy is evaluated by the auxiliary function. The presence of a diagonal line in $\mathbf{A}^{C\leftrightarrow P}$, seen in Figure 5b suggests strong alignment between the code and driving policy domain. As observed again, the "highway" environment showcases a prominent dark red diagonal that is surrounded by darker blue areas. This implies that the policies are highly specialized and are, therefore, behaviorally diverse. Finally, to verify π^P's alignment with $\mathcal{L}$, human annotators assess the behavior's emergence rate across 30 rollouts. High emergence rates and visualizations in Figures 3 and 4 indicate consistent behavior adherence and further demonstrate this alignment.

Methods	Jensen-Shannon Divergence (IQR) ↑		
	Intersection	Merge	Highway
Random Policy (6 skills)	0.1197 (0.0019)	0.2297 (0.0040)	0.2515 (0.0022)
Random Policy (30 skills)	0.1385 (0.0014)	0.2250 (0.0084)	0.3033 (0.0007)
Human Expert (5 skills)	0.1686 (0.0313)	0.2595 (0.0239)	**0.3686 (0.0442)**
DIAYN (6 skills)	0.0107 (0.0062)	0.0152 (0.0038)	0.0254 (0.0021)
DIAYN (18 skills)	0.0163 (0.0039)	0.0211 (0.0058)	0.0319 (0.0014)
DIAYN (36 skills)	0.0181 (0.0079)	0.0083 (0.0027)	0.0195 (0.0067)
Ours (6 skills)	**0.1845 (0.1085)**	**0.3397 (0.0523)**	0.3039 (0.0729)

Table 1. Trajectory diversity using JSD

Driving Policy Diversity. We use an existing metric introduced in [18] to measure the trajectory diversity via the Jensen-Shannon Divergence (JSD). We report the median JSD across all agents on 30 different seedings for each map in Table 1. To contextualize these findings, we benchmarked against three different baselines: random behaviors, unsupervised skill acquisition algorithms, and driving policies trained on expert-crafted reward functions [14]. Random behaviors were generated by defining π^P as a uniform distribution at varied seeding. Next, we compared ***T2D*** to Diversity is All You Need (DIAYN) [9], an established unsupervised skill acquisition algorithm. We adapted DIAYN into a multi-agent setting and trained for 3 different skill counts per map (6, 18, and 36). We report the median $\mathcal{D}^P$ in Table 1. Our results indicate that ***T2D*** surpass random policies and DIAYN-generated policies across all tested scenarios. Notably, ***T2D*** exhibits the highest $\mathcal{D}^P$ in "merge" scenarios. Even in "intersection" and "highway" scenarios, our approach demonstrates competitive diversity, only trailing the human expert in "highway" scenarios.

4. CONCLUSION

In our work, we introduce Text-to-Drive to generate diverse driving behaviors from natural language descriptions. With our knowledge-driven approach, we demonstrate that T2D generates more diverse trajectories compared to other baselines and offers a natural language interface that allows for incorporating human preference.

References

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

[2] Alexander Amini, Tsun-Hsuan Wang, Igor Gilitschenski, Wilko Schwarting, Zhijian Liu, Song Han, Sertac Karaman, and Daniela Rus. Vista 2.0: An open, data-driven simulator for multimodal sensing and policy learning for autonomous vehicles. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2419–2426. IEEE, 2022. 1

[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, 2021. 1

[4] Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric Wolff, Alex Lang, Luke Fletcher, Oscar Beijbom, and Sammy Omari. Nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles, 2022. 1

[5] Yulong Cao, Boris Ivanovic, Chaowei Xiao, and Marco Pavone. Reinforcement learning with human feedback for realistic traffic simulation, 2023. 1

[6] Wei-Jer Chang, Francesco Pittaluga, Masayoshi Tomizuka, Wei Zhan, and Manmohan Chandraker. Controllable safety-critical closed-loop traffic simulation via guided diffusion, 2023. 1

[7] Wenhao Ding, Yulong Cao, Ding Zhao, Chaowei Xiao, and Marco Pavone. Realgen: Retrieval augmented generation for controllable traffic scenarios, 2023. 1

[8] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017. 1

[9] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function, 2018. 4

[10] Lan Feng, Quanyi Li, Zhenghao Peng, Shuhan Tan, and Bolei Zhou. Trafficgen: Learning to generate diverse and realistic traffic scenarios. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3567–3575. IEEE, 2023. 1

[11] Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, and Ming Zhou. Codebert: A pre-trained model for programming and natural languages, 2020. 3

[12] Daniel J. Fremont, Tommaso Dreossi, Shromona Ghosh, Xiangyu Yue, Alberto L. Sangiovanni-Vincentelli, and Sanjit A. Seshia. Scenic: a language for scenario specification and scene generation. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*. ACM, 2019. 1

[13] 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, 2023. 1

[14] Edouard Leurent. An environment for autonomous driving decision-making. https://github.com/eleurent/highway-env, 2018. 4

[15] Quanyi Li, Zhenghao Peng, Lan Feng, Qihang Zhang, Zhenghai Xue, and Bolei Zhou. Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 1

[16] Quanyi Li, Zhenghao Peng, Lan Feng, Zhizheng Liu, Chenda Duan, Wenjie Mo, and Bolei Zhou. Scenarionet: Open-source platform for large-scale traffic scenario simulation and modeling. *Advances in Neural Information Processing Systems*, 2023. 1

[17] Yiyi Liao, Jun Xie, and Andreas Geiger. Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d, 2022. 1

[18] Andrei Lupu, Brandon Cui, Hengyuan Hu, and Jakob Foerster. Trajectory diversity for zero-shot coordination. In *Proceedings of the 38th International Conference on Machine Learning*, pages 7204–7213. PMLR, 2021. 4

[19] Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models, 2023. 1

[20] Sivabalan Manivasagam, Shenlong Wang, Kelvin Wong, Wenyuan Zeng, Mikita Sazanovich, Shuhan Tan, Bin Yang, Wei-Chiu Ma, and Raquel Urtasun. Lidarsim: Realistic lidar simulation by leveraging the real world, 2020. 1

[21] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning, 2016. 3

[22] NVIDIA. DRIVE Sim: Simulation for Self-Driving Cars. https://developer.nvidia.com/drive/simulation, 2022. Accessed: 2024-4-5. 1

[23] Davis Rempe, Jonah Philion, Leonidas J. Guibas, Sanja Fidler, and Or Litany. Generating useful accident-prone driving scenarios via a learned traffic prior, 2022. 1

[24] Junha Roh, Chris Paxton, Andrzej Pronobis, Ali Farhadi, and Dieter Fox. Conditional driving from natural language instructions, 2019. 1

[25] Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Ping Luo, Andreas Geiger, and Hongyang Li. Drivelm: Driving with graph visual question answering, 2023. 1

[26] Jiayang Song, Zhehua Zhou, Jiawei Liu, Chunrong Fang, Zhan Shu, and Lei Ma. Self-refined large language model as automated reward function designer for deep reinforcement learning in robotics, 2023. 1

[27] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Sheng Zhao,

Shuyang Cheng, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset, 2020. 1

[28] Simon Suo, Sebastian Regalado, Sergio Casas, and Raquel Urtasun. Trafficsim: Learning to simulate realistic multi-agent behaviors, 2021. 1

[29] Shuhan Tan, Kelvin Wong, Shenlong Wang, Sivabalan Manivasagam, Mengye Ren, and Raquel Urtasun. Scenegen: Learning to generate realistic traffic scenes, 2021. 1

[30] Shuhan Tan, Boris Ivanovic, Xinshuo Weng, Marco Pavone, and Philipp Kraehenbuehl. Language conditioned traffic generation, 2023. 1

[31] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world, 2017. 1

[32] Danfei Xu, Yuxiao Chen, Boris Ivanovic, and Marco Pavone. Bits: Bi-level imitation for traffic simulation, 2022. 1

[33] Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montse Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, Brian Ichter, Ted Xiao, Peng Xu, Andy Zeng, Tingnan Zhang, Nicolas Heess, Dorsa Sadigh, Jie Tan, Yuval Tassa, and Fei Xia. Language to rewards for robotic skill synthesis. *Arxiv preprint arXiv:2306.08647*, 2023. 1

[34] 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, 2023. 1