PGDrive: Procedural Generation of Driving Environments for Generalization

Quanyi Li*[1], Zhenghao Peng*[2], Qihang Zhang[3,4], Chunxiao Liu[3], Bolei Zhou[2]

1. Introduction

A dilemma in prototyping autonomous driving systems is that, due to public safety concerns, it is impossible to directly develop, train, and test the self driving systems in the physical world [13]. To overcome the dilemma, a common practice is to prototype and validate the driving system in simulators. Many realistic driving simulators have been proposed. CARLA [4] and SUMMIT [1] contain driving scenarios with realistic visual appearances. Flow [14], Duckietown [2], and Highway-env [7] either focus on learning high-level decisions or only contain simplistic driving scenarios. However, it's notorious that neural networks can overfit training data easily [15, 3], thus the driving agent trained on a fixed set of scenarios in the simulators might have difficulty generalizing to new scenarios. Furthermore, the existing simulators lack sufficiently diverse designs of the maps and scenarios to evaluate such issues. For example, CARLA consists of only about ten manually designed towns, which is hard to modify, while Highway-env contains six typical traffic scenarios. Training deep neural networks on such simulators might lead to overfitting and poor generalization of learned driving policy.

To evaluate and improve the generalization of end-to-end driving, we introduce PGDrive, an open-ended and highly configurable driving simulator. The key feature of PGDrive is Procedural Generation (PG) [12, 10]. Unlimited number of diverse driving scenes can be generated with the help of our abstraction on driving scene as well as the proposed PG algorithm. PGDrive is highly configurable, allowing users to customize settings such as the traffic flow and the vehicles' dynamics models. Besides, fine-grained RGB camera, depth camera, Lidar data generation, and realistic 3D kinematics simulation are also supported. To support fast prototyping of driving system, PGDrive is designed to be lightweight and highly efficient, compared to other simulators which are expensive to run or difficult to install. Single PGDrive process can run up to 1000 simulation steps per second without rendering on a PC and can be easily paralleled to further boost efficiency. We open-source PGDrive at https://decisionforce.github.io/pgdrive.

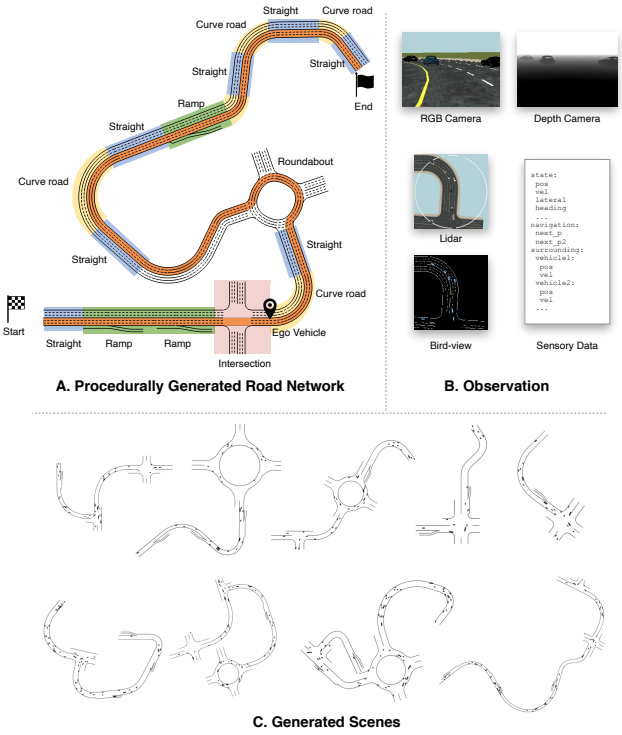

Figure 1. **A.** A procedurally generated road network built from elementary road blocks. **B.** Multi-modal observations provided by PGDrive, including Lidar-like cloud points, RGB / depth camera, bird-view semantic map and scalar sensory data. **C.** Procedurally generated scenes with different number of road blocks.

Based on the proposed PGDrive simulator, we study the generalization of learning-based driving systems. We first validate that the agent trained on a small fixed set of maps generalizes poorly to new scenarios. Then we show that training with more environments, the learned reinforcement learning (RL) agents can generalize better to unseen scenarios in terms of less collision rate and traffic violation. It

*Quanyi Li and Zhenghao Peng contribute equally to this work.
[1] Quanyi Li is with Centre for Perceptual and Interactive Intelligence, Hong Kong SAR. qyli@cpii.hk
[2] Bolei Zhou and Zhenghao Peng are with the Department of Information Engineering, the Chinese University of Hong Kong, Hong Kong SAR. {bzhou, pengzh}@ie.cuhk.edu.hk
[3] Chunxiao Liu and Qihang Zhang are with SenseTime Group Limited. liuchunxiao@senseauto.com
[4] Qihang Zhang is with the College of Computer Sciende and Technology, Zhejiang University. qh_zhang@zju.edu.cn

demonstrates the benefit of procedural generations brought by PGDrive. We also raise a concerning issue called safety generalization. We find that the overfitting of safety performance also exists, and the policy trained with a limited number of scenes yields frequent crashes. We open-source PGDrive to facilitate the research of end-to-end driving.

2. PGDrive Simulator

The key function of PGDrive is to generate unlimited number of diverse scenes. We decompose a driving scene into several components: (1) the *road network*, which consists a set of *road blocks* and the interconnection between them, and (2) the *traffic flow*, containing a set of *traffic vehicles* and a *target vehicle* that actuated by external policy.

We define seven typical types of road block: Straight, Ramp, Fork, Roundabout, Curve, T-Intersection and Intersection. Each block preserves properties like lanes, *spawn points* for generating new traffic vehicles, and *sockets*, namely the exits and entrances, that can be used to interconnect other blocks. We use the Procedural Generation (PG) technique to automatically select and assemble these blocks into one driving scene, as shown in Fig. 1**A**. We propose a search-based PG algorithm *Block Incremental Generation (BIG)*, which recursively appends block to the existing road network if feasible and reverts last block otherwise. After generating a road network, the initial traffic flow with given density (defined by the number of traffic vehicles per lane per 10 meters) is attached to the static road network to complete the scene generation.

PGDrive is implemented based on Panda3D [5] and Bullet Engine and provides various optional forms of observation to feed the external policy. A large number of traffic vehicles cruise in the scene from their spawn points to the randomly assigned destinations. The cruise behavior is powered by the IDM (Intelligent Driving model) with preset target speed. We also implement emergent stopping and lane changing behaviors for traffic vehicles with pre-defined heuristic. Many driving applications can be built upon the proposed PGDrive simulator, such as the benchmarking the generalization ability with spited sets of training and test scenes, multi-agent RL environment by replacing the traffic vehicles with controllable vehicles, or benchmarking safety performance by evaluating collision of dangerous behavior.

3. Experiment

Based on the proposed PGDrive simulator, we conduct experiments to reveal the overfitting issue in the end-to-end driving agent, and show the improvement of the generalization brought by the PG technique. We split the generated scenes into the exclusive training set and test set. We evaluate the agent for multiple episodes and define the ratio of episodes where the agent arrives the destination as *success rate*, which will be reported in experiments instead of episodic reward.

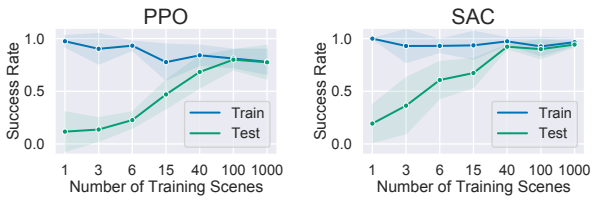

Figure 2. The main experimental results demonstrate the generalization improvement brought by increased training diversity.

Results on Generalization We train the agents with two popular RL algorithms respectively, PPO [11] and SAC [6], based on the implementation in RLLib [8]. As shown in Fig. 2, the result of improved generalization is observed in the agents trained from both RL algorithms: First, the overfitting happens if the agent is not trained with a sufficiently large training set. Second, the generalization ability of agents can be greatly improved, in term of the performance gap between training and test, if trained in more environments. Besides, SAC shows better generalization compared to PPO. The experimental results clearly show that increasing the diversity of training environments can significantly increase the generalization of RL agents [3], which validates the strength of the PG mechanism in PGDrive.

Safety Generalization. We randomly place obstacles in the road and provides $+1$ cost if collision happens. We test the *reward shaping method*, which assigns the crash penalty as negative reward, as well as the *Lagrangian method* [9] which is designed for constrained MDP. When trained with few scenes, both methods achieve high test cost. Increasing the diversity of training scenes reduces test cost and shows better safety generalization. Lagrangian method outperforms reward shaping method and reduces the test cost by around 50%, but still yield poor safety performance when trained with limited scenes. This experimental result, for the first time, reveals the overfitting of safety.

Other factors relevant to generalization. We find that randomizing the traffic density can improve the generalization of agents under the "unseen traffic flow". The agent trained with varying traffic density in range $[0.0, 0.4]$ consistently outperforms the agent trained in the fixed density 0.1. Besides, the friction coefficient is also a critical factor that influences generalization. The agent trained in low friction can successfully drive in environments of both low and high friction, while one trained in high friction fails to drive in low friction environment. We further conduct an experiment to show that an agent specialized on solving all types of blocks in separated single-block environments can not solve a complex driving map composed of multiple blocks.

References

[1] Panpan Cai, Yiyuan Lee, Yuanfu Luo, and David Hsu. Summit: A simulator for urban driving in massive mixed traffic. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4023–4029. IEEE, 2020. 1

[2] Maxime Chevalier-Boisvert, Florian Golemo, Yanjun Cao, Bhairav Mehta, and Liam Paull. Duckietown environments for openai gym. https://github.com/duckietown/gym-duckietown, 2018. 1

[3] Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*, pages 1282–1289. PMLR, 2019. 1, 2

[4] 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

[5] Mike Goslin and Mark R Mine. The panda3d graphics engine. *Computer*, 37(10):112–114, 2004. 2

[6] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pages 1861–1870, 2018. 2

[7] Edouard Leurent. An environment for autonomous driving decision-making. https://github.com/eleurent/highway-env, 2018. 1

[8] Eric Liang, Richard Liaw, Robert Nishihara, Philipp Moritz, Roy Fox, Ken Goldberg, Joseph Gonzalez, Michael Jordan, and Ion Stoica. Rllib: Abstractions for distributed reinforcement learning. In *International Conference on Machine Learning*, pages 3053–3062, 2018. 2

[9] Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. *openai*, 2019. 2

[10] Sebastian Risi and Julian Togelius. Increasing generality in machine learning through procedural content generation. *Nature Machine Intelligence*, pages 1–9, 2020. 1

[11] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 2

[12] Noor Shaker, Julian Togelius, and Mark J Nelson. *Procedural content generation in games*. Springer, 2016. 1

[13] Ardi Tampuu, Maksym Semikin, Naveed Muhammad, Dmytro Fishman, and Tambet Matiisen. A survey of end-to-end driving: Architectures and training methods. *arXiv preprint arXiv:2003.06404*, 2020. 1

[14] Eugene Vinitsky, Aboudy Kreidieh, Luc Le Flem, Nishant Kheterpal, Kathy Jang, Cathy Wu, Fangyu Wu, Richard Liaw, Eric Liang, and Alexandre M Bayen. Benchmarks for reinforcement learning in mixed-autonomy traffic. In *Conference on Robot Learning*, pages 399–409, 2018. 1

[15] Chiyuan Zhang, Oriol Vinyals, Remi Munos, and Samy Bengio. A study on overfitting in deep reinforcement learning. *arXiv preprint arXiv:1804.06893*, 2018. 1