Supplementary Material

Title

TrafficGamer: Reliable and Flexible Traffic Simulation for Safety-Critical Scenarios with Game-Theoretic Oracles

Authors

Guanren Qiao[1], Guorui Quan[2], Jiawei Yu[1], Shujun Jia[3], Guiliang Liu[1*]

Affiliations

[1]The Chinese University of Hong Kong, Shenzhen
[2]The University of Manchester
[3]Shenyang MXNavi Co.,Ltd.
*Corresponding Author, liuguiliang@cuhk.edu.cn

1. Supplementary information for experiments

a. Dataset

The Argoverse 2 dataset consists of 250,000 non-overlapping scenarios [1] (80/10/10 train/val/test random splits). Each scenario contains its own HD Map with 3D lane and crosswalk geometry. The first 5s of each scenario is denoted as the *observed* window, while the subsequent 6s is denoted as the *forecasted* horizon. Each scenario has a single track as the *focal agent*. Focal tracks are guaranteed to be fully observed throughout the scenario and have been specifically selected to maximize *interesting interactions* with map features and other nearby actors.

For each scenario of the Waymo open motion dataset (WOMD) 1.2.0, the first 3s is denoted as the *observed* window, while the subsequent 8s is denoted as the *forecasted* horizon [2]. Other settings are similar to the Argoverse 2 dataset.

b. Experiment settings

We calculate the fidelity-related metrics with all selected scenarios to validate the statistical realism of the TrafficGamer. We use 123, 321, and 666 as the experimental seeds, and we present the mean ± standard deviation (std) for each evaluated algorithm.

c. Baseline settings

Our baseline methods include: 1) QCNet [3] introduces a query-centric paradigm for scene encoding, which enables the reuse of past computations by learning representations independent of the global spacetime coordinate system and first employ anchor-free queries to generate trajectory proposals in a recurrent fashion, which allows the model to utilize different scene contexts when decoding waypoints at different horizons. A refinement module then takes the trajectory proposals as anchors and leverages anchor-based queries to refine the trajectories further. 2) GameFormer [4] incorporates a Transformer encoder, which effectively models the relationships between scene elements, alongside a novel hierarchical Transformer decoder structure. At each decoding level, the decoder utilizes the prediction outcomes from the previous level, in addition to the shared environmental context, to iteratively refine the interaction process. 3) Multi-agent Proximal Policy Optimization (MAPPO) [5] adapts PPO [6] to solve multi-agent cooperation or competition problems based on centralized training decentralized execution. In our work, we also use MAPPO to fine-tune our world model as one of the baselines.

d. Performance of world model

Since our task is based on trajectory prediction, we introduce trajectory prediction-related metrics to evaluate the world model. For the Argoverse 2 dataset, we adopt metrics including:

- minimum Average Displacement Error ($minADE_K$): calculates the l_2 distance in meters between the ground-truth trajectory and the best of K predicted trajectories as an average of all future time steps.
- minimum Final Displacement Error ($minFDE_K$): concerns the prediction error at the final time step to emphasize long-term performance.
- Brier-minimum Final Displacement Error ($b-minFDE_K$): adds $(1-\hat{\pi})^2$ to the final-step error, where $\hat{\pi}$ denotes the best-predicted trajectory's probability score that the model assigns.
- Miss Rate(MR_K): counts the ratio of cases where $minFDE_K$ exceeds 2 meters.

For the WOMD, we adopt metrics including:

- minimum Average Displacement Error ($minADE_K$): computes the mean of the l_2 norm between the ground truth and the closest prediction.
- minimum Final Displacement Error ($minFDE_K$): equivalent to evaluating the $minADE$ metric at a single time step T.

- Overlap Rate (OR): computed by taking the highest confidence prediction from each set of predictions for each object. If any of these predicted agent trajectories overlap at any time with any other objects that were visible at the prediction time step (compared at each time step up to T) it is considered a single overlap. The overlap rate in this challenge is computed as the total number of overlaps divided by the total number of objects predicted.
- Miss Rate (MR_K): defined as the state when none of the individual K predictions for an object are within a given lateral and longitudinal threshold of the ground truth trajectory at a given time T.
- Mean average precision (mAP): computes the area under the precision-recall curve by applying confidence score thresholds across a validation set and using the definition of Miss Rate above to define true positives, false positives, etc.
- Soft mean average precision ($Soft - mAP$): handles multiple matching predictions for a given trajectory. Additional matching predictions are ignored and are not penalized as part of the metric computation.

K is selected as 6. If a model outputs more than K trajectories, only the predictions with the top-K probability scores are considered during evaluation. The performance of the generative world model can be seen in Table 1 and Table 2.

e. Training details

The hidden feature dimension is 128. The number of heads in the multi-head attention operator is 8. Layer normalization (LayerNorm) is used in Multilayer Perceptron (MLP)s and attention layers [7]. We adopt the AdamW optimizer [8] to train the world model in an end-to-end manner for 64 epochs with a batch size of 32, a dropout rate of 0.1, and a weight decay coefficient of 1×10^{-4}. Due to the larger scale of WOMD, we train models for 64 epochs with a batch size of 16. The learning rate is initialized to 5×10^{-4} and is decayed using the cosine annealing scheduler [9]. We use the cross-entropy loss as the classification loss of the training world model to optimize the mixing coefficients. For the implementation of policy and value functions, we use two-layer MLP, one-layer LayerNorm, and one-layer Relu to construct them [10]. For the estimation of observation density, we design an observation density estimator to compute a density using a Normaling Flow-based method [11]. This estimator is implemented by stacking multiple MADE layers [12]. For the section of risk-sensitivity, we apply SplineDQN [13], which are made up of two-layer MLP, LayerNorm and Relu to estimate the cost value of observations. We showcase all parameters of the reinforcement learning (RL) section of TrafficGamer (see Table 3).

For the Argoverse 2 dataset, the training process for the world model utilizes 180GB of GPU memory across 8 RTX 4090s and typically runs around 48 hours. Fine-tuning a scenario requires 1 RTX 3090 and 5-8 hours of running. For the WOMD, the training process for the world model utilizes 480GB of GPU memory across 8 A100s and typically runs around 144 hours. Fine-tuning a scenario requires 1 RTX 3090 and 8-15 hours of running.

2. Supplementary information for methods

a. Reward function and cost function

For reward function, we design a piecewise function:

$$R_1(\cdot) = -\frac{\|d_{total} - d_{travel}\|_1}{c_1} \tag{1}$$

where d_{total} is the distance to the endpoint, d_{travel} means the distance you have traveled, and c_1 is a constant. $R_2(\tau)$ detects whether the vehicle will drift off the lane. If it does, $R_2(\cdot) = -c_2$. $R_3(\tau)$ determines whether the vehicle has collided. $R_3(\cdot) = -c_3$ if a collision occurs. We set $c_1 = 1, c_2 = c_3 = 10$. $R_4(\tau)$ defines if the velocity speed is higher than the limited maximum velocity. $R(\tau)$ is ultimately the sum of rewards r_1, r_2, r_3 and r_4.
For the cost function, we design a binary classification function:

$$C(\cdot) = \|p_i - p_j\|_2 \qquad (i \neq j) \tag{2}$$

where i or j is the agent and p denoted as the position of agent. If $c \leq \epsilon$, the cost value is 1, where ϵ is the distance constraint. Otherwise, the cost value is 0.

3. Supplementary information for results

Argoverse 2 results

a. Performance of other agents in exploitability experiment

Due to space limitations in the paper, we have placed the performance of other agents capturing CCE here. Figure 1 shows that regardless of whether using the first or second calculation method, TrafficGamer consistently estimates CCE better and learns the optimal strategy.

b. Lagrangian parameters variation of other agents

Due to space limitations in the paper, we show the other agents' variation of Lagrangian parameters in some scenarios here. Figure 2 shows that regardless of whether using the first or second calculation method, TrafficGamer consistently estimates CCE better and learns the optimal strategy.

c. Crash rate of different datasets

To accurately assess AV systems, generated scenarios must match real-world accident rates. We evaluate TrafficGamer's ability to produce safe trajectories with crash rate metric. In this work, the crash rate is computed by dividing the number of times steps when collisions occur

by the total number of time steps. Through experiments, we calculate that the collision rate of TrafficGamer is 0 in all scenarios. We draw the same conclusion on the Waymo dataset, so the fidelity can be guaranteed across different datasets.

WOMD results

d. Fidelity verification in WOMD dataset

The TrafficGamer fidelity results using the WOMD dataset are shown in Supplementary Figure 3. From the results, we can find that TrafficGamer can achieve statistical realism.

References

[1] Wilson, B. *et al.* Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *Advances in Neural Information Processing Systems Track on Datasets and Benchmarks 1, (NeurIPS)* (2021).

[2] Ettinger, S. *et al.* Large scale interactive motion forecasting for autonomous driving : The waymo open motion dataset. In *IEEE/CVF International Conference on Computer Vision, (ICCV)* (2021).

[3] Zhou, Z., Wang, J., Li, Y. & Huang, Y. Query-centric trajectory prediction. In *International Conference on Computer Vision and Pattern Recognition, (CVPR)* (2023).

[4] Huang, Z., Liu, H. & Lv, C. Gameformer: Game-theoretic modeling and learning of transformer-based interactive prediction and planning for autonomous driving. In *International Conference on Computer Vision, (ICCV)* (2023).

[5] Yu, C. *et al.* The surprising effectiveness of PPO in cooperative multi-agent games. In *Advances in Neural Information Processing Systems, (NeurIPS)* (2022).

[6] Schulman, J., Wolski, F., Dhariwal, P., Radford, A. & Klimov, O. Proximal policy optimization algorithms. *CoRR* **abs/1707.06347** (2017).

[7] Zhou, Z., Ye, L., Wang, J., Wu, K. & Lu, K. Hivt: Hierarchical vector transformer for multi-agent motion prediction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, (CVPR)* (2022).

[8] Loshchilov, I. & Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations, (ICLR)* (2019).

[9] Loshchilov, I. & Hutter, F. SGDR: stochastic gradient descent with warm restarts. In *International Conference on Learning Representations, (ICLR)* (2017).

[10] Huang, S., Sun, C., Wang, R.-Q. & Pompili, D. Multi-behavior multi-agent reinforcement learning for informed search via offline training. In *International Conference on Distributed Computing in Smart Systems and the Internet of Things, (DCOSS-IoT)* (2024).

[11] Qiao, G., Liu, G., Poupart, P. & Xu, Z. Multi-modal inverse constrained reinforcement learning from a mixture of demonstrations. In *Advances in Neural Information Processing Systems, (NeurIPS)* (2023).

[12] Germain, M., Gregor, K., Murray, I. & Larochelle, H. MADE: masked autoencoder for distribution estimation. In *International Conference on Machine Learning, (ICML)* (2015).

[13] Luo, Y., Liu, G., Duan, H., Schulte, O. & Poupart, P. Distributional reinforcement learning with monotonic splines. In *International Conference on Learning Representations, (ICLR)* (2022).

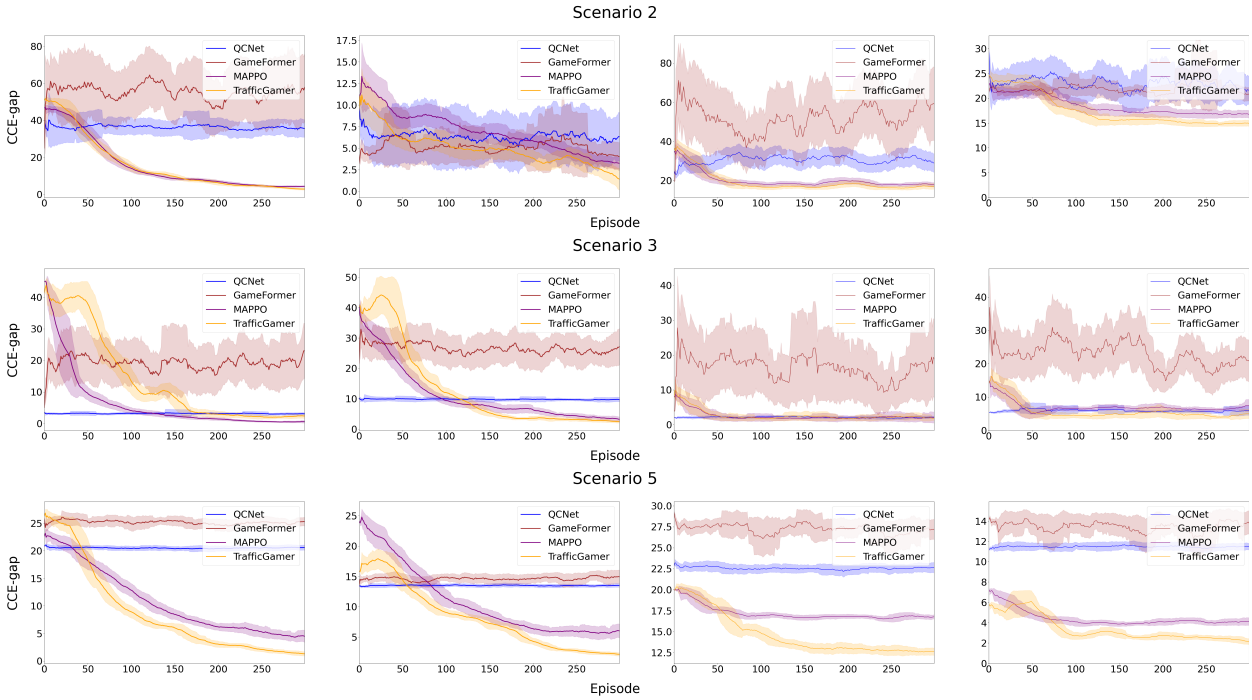

Figure 1: **CCE-gap**, where each row represents a scenario and scenarios 2, 3, and 5 correspond to *Dense-lane intersection*, *Roundabout*, and *T-Junction* respectively. Each column corresponds to one of the agents in the multi-agent environment. The two columns on the left are the results of the first calculation method mentioned in the text, while the two columns on the right are from the second method.

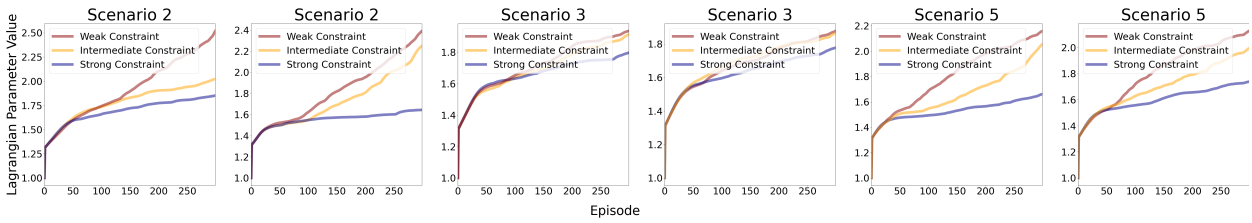

Figure 2: **Lagrangian Parameters Variation**, where scenarios 2, 3, and 5 correspond to *Dense-lane intersection*, *Roundabout*, and *T-Junction* respectively. Each column corresponds to one of the agents in the multi-agent environment.

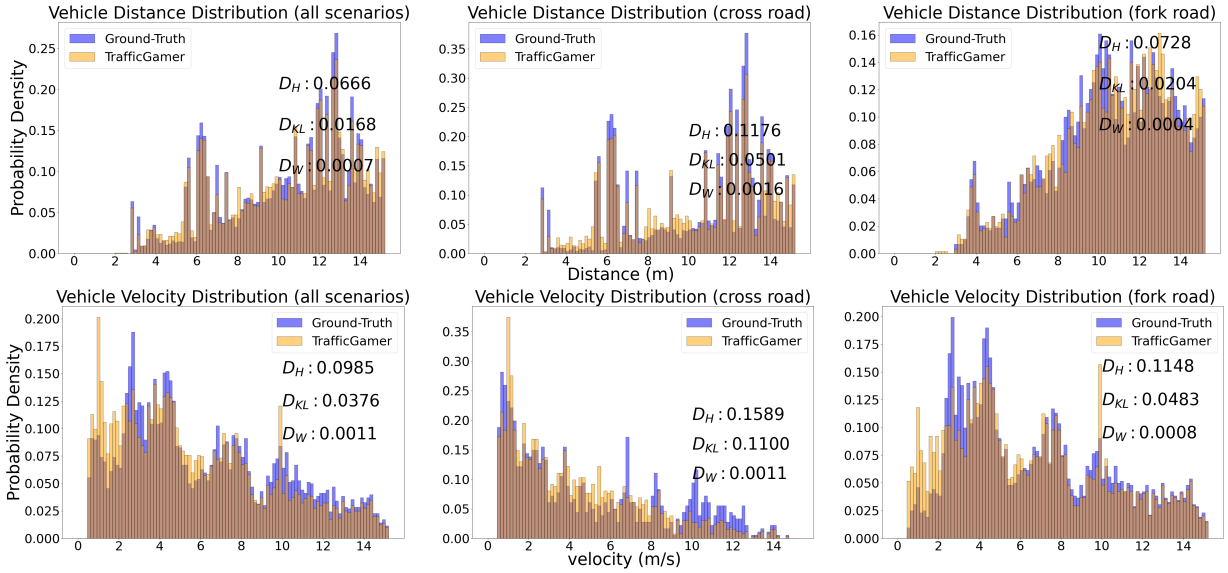

Figure 3: **Statistical realism in WOMD dataset.** From top to bottom are TrafficGamer's vehicle distance and speed distributions. We use various distance metrics to measure the distance between the generated and the real data distribution in three settings: all scenarios, fork road, and cross road respectively.

Table 1: world model performance in Argovese 2

dataset	minFDE (K=6)	minADE (K=6)	MR (K=6)	brier-minFDE (K=6)
validation dataset	1.2529	0.7203	0.1578	1.8637

Table 2: world model performance in WOMD

dataset	minFDE (K=6)	minADE (K=6)	MR (K=6)	OR (K=6)	mAP (K=6)	Soft mAP (K=6)
validation dataset	1.7966	0.8107	0.1596	0.1647	0.4347	0.4387

Table 3: RL section of TrafficGamer parameters table

parameter	Value
seed	123 / 321 / 666
offset	2 / 3 / 5
the number of agents	$n \geq 1$
actor learning rate	5e-5
critic learning rate	1e-4
discounted factor	0.99
gradient clip norm	5e-1
batch size	32
hidden dimension	128
epoch	10
episode	300 / 500
N quantile	64
cost quantile	8 / 32 / 56
τ update	1e-2
distributional value network	SplineDQN
distributional value network learning rate	3e-4
density learning rate	1e-4
risk measure	CVaR
density model coefficient	1e-1
η_1	5e-2
$\frac{1}{\eta_2}$	5e-2
distance constraint	2-8