A Versatile and Efficient Reinforcement Learning Approach for Autonomous Driving

Guan Wang[1*†], **Haoyi Niu**[1*†], **Desheng Zhu**[2†],
Jianming Hu[1], **Xianyuan Zhan**[1‡], **Guyue Zhou**[1]
[1] Tsinghua University [2] China University of Mining and Technology
{wangguan19, nhy22}@mails.tsinghua.edu.cn
{zhanxianyuan,zhouguyue}@air.tsinghua.edu.cn

Abstract

Heated debates continue over the best solution for autonomous driving. The classic modular pipeline is widely adopted in the industry owing to its great interpretability and stability, whereas the fully end-to-end paradigm has demonstrated considerable simplicity and learnability along with the rise of deep learning. As a way of marrying the advantages of both approaches, learning a semantically meaningful representation and then using it in the downstream driving policy learning tasks provides a viable and attractive solution. However, several key challenges remain to be addressed, including identifying the most effective representation, alleviating the sim-to-real generalization issue as well as balancing model training cost. In this study, we propose a versatile and efficient reinforcement learning approach and build a fully functional autonomous vehicle for real-world validation. Considering visual, dynamics and scenario gaps that hinder the sim-to-real transfer, we first standardize *NoGap* Benchmark in CARLA simulator for preliminary validation. In various complicated real-world scenarios, our method also shows great generalizability and superior training efficiency against the competing baselines.

1 Introduction

The past decade has witnessed a surge of research interests in end-to-end autonomous driving systems [Tampuu *et al.*, 2020] driven by imitation learning (IL) [Bojarski *et al.*, 2016; Anderson *et al.*, 2018; Müller *et al.*, 2018; Hecker *et al.*, 2020; Huang *et al.*, 2021; Wang *et al.*, 2021] or reinforcement learning (RL) [Kiran *et al.*, 2021]. Despite its appealing simplicity and learnability endowed with neural networks, most attempts have shown performance or adaptability issues in real-world scenarios due to large visual and dynamics gaps [Rao *et al.*, 2020; Peng *et al.*, 2018] between simulation and real-world environments. These unsolvable issues remind us of the conventional modular pipeline [Levinson *et al.*, 2011] that divides the system into modules for error-tracking and enables vehicles to behave predictably. However, this approach lacks flexibility and leads to tedious human engineering in devising complicated rules and model fine-tuning.

The corresponding characteristics naturally inspire studies to investigate the marriage between end-to-end driving and modular pipeline, such as uncovering a series of intermediate representations as inputs for the driving decision-making model. Affordance learning [Chen *et al.*, 2015; Sun *et al.*, 2019; Sauer *et al.*, 2018] predicts a heavily compressing representation for driving decision-making, e.g. comprising the distances to surrounding vehicles, heading angles, etc. The substantially low dimensionality hinders the downstream controller with brittleness and sensitivity, since little prediction error

[*] Equal contribution.
[†] Work done during internships at Institute for AI Industry Research (AIR), Tsinghua University.
[‡] Corresponding author.

Machine Learning for Autonomous Driving Workshop at the 36th Conference on Neural Information Processing Systems (NeurIPS 2022), New Orleans, USA.

on the representation induces large deviation on the decision output. The image-to-image translation is to transform the simulated images into photo-realistic ones by domain adaptation techniques like CycleGAN [Bewley *et al.*, 2019]. Although this alleviates the sim-to-real generalization issues in IL or RL-based autonomous driving approaches, it does not extract an effective representation to reduce the complexity of the downstream decision-making task. We highlight the necessity of learning a suitable representation with both efficient features and sufficient information. Contrary to the trend of pursuing complex and heavy-weighted solutions, we opt for a simple yet effective approach that offers competitive real-world adaptability, i.e. mapping sensor inputs into an appropriately low-dimensional and semantic-meaningful representation [Huang *et al.*; Müller *et al.*, 2018], which can greatly ease the burden of learning complex driving policies with methods like RL [Lesort *et al.*, 2018].

In this paper, we present a deployed versatile and efficient autonomous driving approach, which contains two steps: 1) **Semantic Perception** copes with the raw input from camera images and uses advanced scene understanding models [Cordts *et al.*, 2016] to perform drivable space and lane boundary identification, which empowers the system with versatility and adaptability since inputs from different domains would be mapped into a canonical representation space. In parallel, the lower-dimensional post-perception outputs reduce the load and fully unleash the potential of the follow-up RL policy training; 2) **Distributed RL** learns driving policies from the scene understanding outputs via a carefully designed RL model with a distributed acceleration training scheme (*OneRL*[4]), which enjoys highly efficient off-policy RL training. Given actions from the learned RL policy, mature PID controllers are used to regulate and control the low-level driving commands.

We evaluate our method against multiple baselines in CARLA simulator [Dosovitskiy *et al.*, 2017] on our proposed *NoGap* benchmark[5] with intentionally introduced visual, dynamics and scenario gaps to demonstrate real-world adaptability. Meanwhile, it is notable that our solution reduces the training time consumption by a fair margin. To underpin real-world validation, this study also makes a major contribution by building a real autonomous vehicle to deploy the proposed method. In the subsequent real-world experiments, our system proves to be capable of generalizing to diverse complicated scenarios with varying road topology and lighting conditions, as well as the presence of obstacles.

2 Related Work

2.1 Modular VS End-to-End Autonomous Driving

Currently, mainstream architectures for autonomous driving stem from either a modular or an end-to-end way. The conventional **modular pipeline** comprises a multitude of sub-systems, such as perception, localization, prediction, planning, and control [Levinson *et al.*, 2011; Yurtsever *et al.*, 2020]. In order to handle as many real-world scenarios as possible, the interoperation of these highly functional modules relies on human-engineered deterministic rules. The first in-depth discussions and analyses of the low-level software that encapsulates these rules are in the famous DARPA Challenge [Thrun *et al.*, 2006; Montemerlo *et al.*, 2008], paving the way for many later studies [Levinson *et al.*, 2011]. The modular pipeline is widely adopted in the industry due to its remarkable interpretability. It endows the overall system with the ability to track down and locate sub-system malfunctions. Nevertheless, modular formalism bears several major drawbacks [Tampuu *et al.*, 2020]. Large amounts of human engineering are involved to fine-tune individual and cross-module configurations. The framework also suffers from severe compounding errors when decomposing the whole system into lots of smaller-scale but interpretable modules, as the perception uncertainty and modeling errors would snowball through the pipeline.

Alternatively, **end-to-end** autonomous driving has been increasingly acknowledged. It is widely accepted that human driving is a behavior reflex task [Tampuu *et al.*, 2020] that requires little high-level reasoning and conscious attention, which harbors the same view with the end-to-end architecture. As a deep learning-based solution, the task-specific end-to-end learning ability brings a great reduction of human engineering efforts. However, many studies [Zablocki *et al.*, 2021; Müller *et al.*, 2018; Xiao *et al.*, 2020] recognize the drawback of lack of interpretability in the development of end-to-end systems due to its black-box learning scheme. To alleviate this limitation, Chen *et al.* [Chen *et al.*, 2021] combine the probabilistic graphical modeling with RL to improve the interpretability for autonomous driving in simulation scenarios. Despite these efforts, weak interpretability remains a fatal disadvantage of the end-to-end approaches against the modular architecture [Xiao *et al.*, 2020].

[4] *OneRL* Library: https://github.com/imoneoi/onerl

[5] *NoGap* Benchmark: https://github.com/imoneoi/carla_env

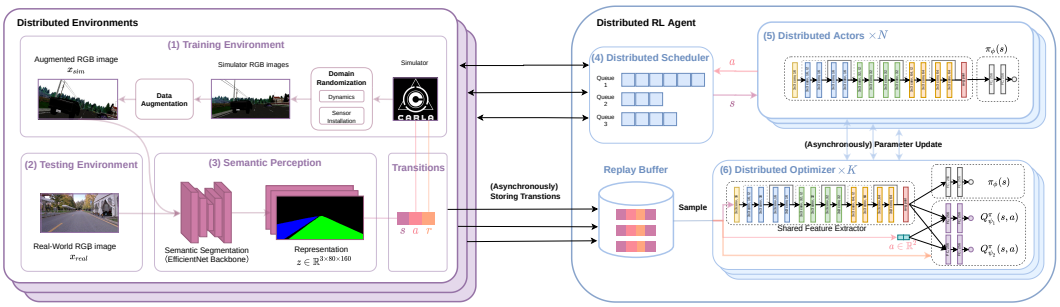


Figure 1: The Overall Architecture

Based on the above comparisons, there is a growing consensus among researchers that combining the merits of both modular and end-to-end formalism could be a promising solution. The raw perceptual inputs are processed through semantic segmentation methods [Cordts *et al.*, 2016], image-translation networks [Bewley *et al.*, 2019], top-down 2D view synthesizing [Bansal *et al.*, 2018], vectorized representation [Scheel *et al.*, 2022], or online mapping and dynamic state representation [Casas *et al.*, 2021]. The intermediate outputs are then used for waypoints planning [Müller *et al.*, 2018] or control decision learning [Behl *et al.*, 2020] in an end-to-end manner. Despite following the philosophy of this combined architecture that alleviates the drawbacks of both approaches to offer enhanced learnability, robustness, and even transferability, we intend to find a minimalist solution instead of persuing complex ones like other works.

2.2 Imitation Learning VS Reinforcement Learning

In end-to-end paradigm, **IL** [Bojarski *et al.*, 2016; Anderson *et al.*, 2018; Müller *et al.*, 2018; Hecker *et al.*, 2020; Huang *et al.*, 2021; Wang *et al.*, 2021] has been identified as a practical paradigm for autonomous driving. It uses a supervised learning scheme that imitates the human experts' demonstrations using algorithms like behavior cloning (BC) [Behl *et al.*, 2020; Prakash *et al.*, 2020; Chen *et al.*, 2020]. IL is known to have several major disadvantages [Tampuu *et al.*, 2020]. First, since most IL algorithms perform supervised learning on offline training datasets, when facing unseen scenarios during closed-loop testing, serious *distribution shift* problems [Ross *et al.*, 2011] take place and the agent has no idea what to do. Second, *data bias* [Codevilla *et al.*, 2019] deeply hurts the generalizability of IL policy, as the training process pays little attention to rare and risky scenarios, namely the long-tailed problem in self-driving [Mao *et al.*, 2021]. Finally, *causal confusion* [Haan *et al.*, 2019] is another problem that occurs in IL, as it performs pure data fitting and handles spurious correlations in data badly. Due to the above limitations, typical IL models can succeed in simple tasks like lane following but underperform in more complicated and rare traffic events. To improve the performance in harder scenarios, conditional imitation learning (CIL) [Codevilla *et al.*, 2018; Xiao *et al.*, 2020; Hawke *et al.*, 2020] introduces a latent state to fully explain the data, thus resulting in a better model expressiveness. However, due to the lack of the ability to perform long-term predictive "reasoning", it still has some safety issues during closed-loop testing [Hawke *et al.*, 2020].

Due to the ability to solve complex tasks as well as perform long-term optimization, **RL** has become another popular choice for investigating end-to-end autonomous driving [Huang *et al.*, 2020; Kiran *et al.*, 2021; Osiński *et al.*, 2020; Liang *et al.*, 2018]. RL learns how to map observations to optimized actions by maximizing expected cumulative reward [Sutton *et al.*, 1998], and can obtain strong policies in simulators without real-world labels [Pan *et al.*, 2017]. Although RL has lower data efficiency than IL, it could be easily implemented in some high-fidelity simulators, e.g. CARLA [Dosovitskiy *et al.*, 2017], where agents can explore in more diverse scenarios. This interactive learning ability resolves the distribution shift, data bias, and causal confusion issues in IL. However, RL is also much harder to train with high-dimensional states. An approach with a dedicated state representation can effectively ease the burden of RL, and also lead to more stable policies. Furthermore, due to the interactive learning nature of RL and its reliance on a simulator, considerable efforts need to be taken to properly address the sim-to-real issue in a deployable autonomous driving system.

3 Methodology

The overall architecture is depicted in Figure 1, decoupling the end-to-end RL paradigm into semantic perception, RL-based decision making, and an additional controller for real-world deployment: **(1) Semantic Perception**: we perform drivable space and lane boundary estimation efficiently based on the input image x from the monocular camera, as supervised input state-encoder. **(2) Distributed RL-based Decision Making**: using the learned representation z, it learns the optimized high-level vehicle control actions $\boldsymbol{a}$, consisting of throttle τ and steering angle θ, with a carefully designed fully distributed RL infrastructure, boosting data and computation efficiency. Moreover, domain generalization methods are applied to improve sim-to-real transfer performance. Finally, the controller maps high-level actions $\boldsymbol{a}$ to low-level vehicle control commands according to the vehicle physical properties using PID controllers.

3.1 Semantic Perception

We choose the results of drivable area segmentation and lane boundary identification [Yu *et al.*, 2020] as the intermediate representation z, as it is an effective way to represent road situations from monocular RGB images and is widely used in autonomous driving systems. It labels every pixel as drivable (lane or area currently occupied by ego-vehicle), alternatively drivable (requires a lane-change), or non-drivable (blocked by obstacles) space.

Crucially, this procedure acts as a powerful supervised state-encoder that maps camera-captured RGB images from simulation x_{sim} or real world x_{real} to a domain-agnostic representation z, effectively bridging the sim-to-real visual gap. Compared to unsupervised state-encoders like image translation networks adopted by [Bewley *et al.*, 2019], semantic segmentation yields low-dimensional and human-readable intermediate representations, containing not only efficient but also sufficient information needed for RL. Notably, this disentanglement of redundant features substantially permits versatility and interpretability that facilitate downstream policy training and transfer.

Our semantic segmentation model utilizes atrous convolution [Chen *et al.*, 2017] to identify road objects with different scales, along with encoder-decoder architecture [Chen *et al.*, 2018] to capture fine-grained details like lane lines. We introduce the light-weighted EfficientNet [Tan and Le, 2019] backbone network to replace the large ResNet101 backbone in previous methods [Ronneberger *et al.*, 2015; Chen *et al.*, 2018], to achieve better real-time performance on onboard computer (see Table 1 for details). Various data augmentation methods (cropping, random noise, perspective transformation, etc.) are used to further improve robustness to different environments and lighting conditions.

3.2 Distributed Reinforcement Learning

Our decision making model is an RL agent trained in the CARLA simulator [Dosovitskiy *et al.*, 2017] using the representation obtained through semantic segmentation as inputs. In complex RL tasks like autonomous driving, shrinking the extensive amount of computation time with limited resources is a crying demand. To this end, we tailor a fully distributed scheme (illustrated in Figure 1) for sample efficient off-policy RL training, with domain generalization and specially designed network architecture, to efficiently exploit system resources and improve final performance.

3.2.1 Problem Formulation

We formulate the driving decision-making problem as a Markov Decision Process (MDP) defined by a tuple $(S, A, r, T, \rho, \gamma)$, where $T(s'|s, a)$ denotes the transition dynamics, ρ is the initial state distribution and $\gamma \in (0, 1)$ is the discount factor. In our problem, S, A and r are defined as follows: **(1) States** S: we use the drivable area segmentation z from semantic perception, concatenated with the vehicle speed v as the state. **(2) Actions** A: We consider the steering angle θ and throttle τ of vehicle as actions $a = [\theta, \tau]$, where $\theta, \tau \in [-1, 1]$. **(3) Reward function** r: The design of our reward function, detailed in Appendix A, involves concerns about four aspects: speed control r_{speed}, lane center keeping r_{center}, heading direction alignment r_{heading} as well as collision and undesired lane crossing penalty r_{penalty}. We solve the MDP problem using RL, which aims at learning a parameterized policy π_ϕ to maximize following expected cumulative discounted reward:

$$\mathbb{E}_{s_0 \sim \rho, a_t \sim \pi_\phi(s), s_{t+1} \sim T(s_{t+1}|s_t, a_t)}[\sum_{t=0}^{\infty} \gamma^t r_t] \tag{1}$$

In the commonly used actor-critic RL paradigm, one optimizes the policy π_ϕ by alternatively maximizing a value function $Q_\psi(s, a)$ to approximate the cumulative return, which is learned by

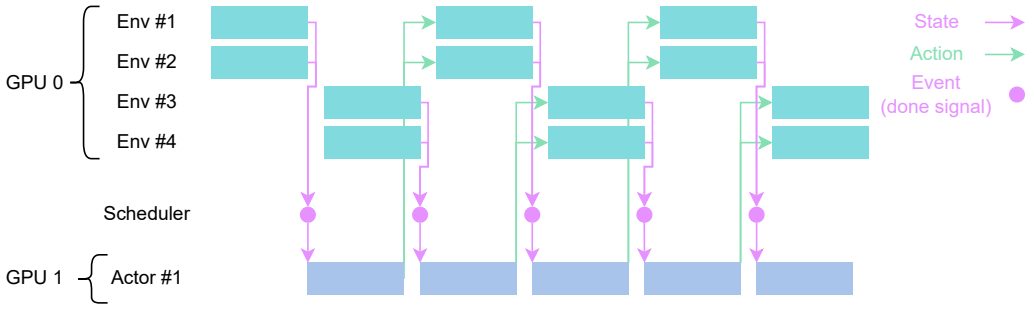


Figure 2: Distributed pipeline execution

minimizing the squared Bellman error:

$$J_Q(\psi) = \mathbb{E}_{s,a,s',a' \sim \pi_\phi(s)} \left[Q^\pi_\psi(s,a) - (r + \gamma Q^\pi_{\psi'}(s',a')) \right]^2 \tag{2}$$

where $Q^\pi_{\psi'}(s,a)$ is target Q-function, typically served as a delayed copy of current Q-function.

3.2.2 Distributed Off-policy RL Architecture

Learning a driving policy using RL can be rather costly, we develop a fully distributed event-driven actor-critic RL Library *OneRL*, which leverages multiple simulation environments, actor policies and optimizer nodes to maximally accelerate RL training.

Specifically, we use N_1 training environments and N_2 actor policies (Figure 1(1, 5)) for parallel data collection, all of which are implemented as independent nodes. An event scheduler (Figure 1(4)) is used to pipeline the execution of these environments and actors, making full use of computing resources. The generated trajectories are then collected asynchronously by the replay buffer to be consumed by the off-policy RL algorithm to enhance data efficiency.

For the learning process, K optimizer nodes (Figure 1(6)) are instantiated to asynchronously update all the actor policies and Q networks. Each optimizer node holds the same copy of all the N_2 policies, and also contains M Q-networks (contain both current and multiple target Q-networks), leading to a total of $K \times M$ Q-networks in our training process. The optimizers compute gradients of policies and Q-networks using batches of transitions asynchronously sampled from the replay buffer. Distributed all-reduce is used to synchronize gradient shards for optimization. The final step towards completing the training loop is to asynchronously update the policies (Figure 1(5)) with the snapshots of the policy network weights in optimizers.

Contrary to conventional distributed architecture like IMPALA [Espeholt *et al.*, 2018] and Ape-X [Horgan *et al.*, 2018], which maintains the environment and actor in the same node, we devise a scheduler to separate all the components (environment, actor, replay buffer and optimizer), making our architecture fully distributed. In addition, these architectures only access to CPU for single-step environment simulation and actor network inference. Our distributed scheduler batches states for GPU pipelined inference (see Figure 2) to minimize latency. This also enables the flexibility to operate actors and environments on different devices (like CPU for physics simulation, or GPU for scene rendering). Our architecture also supports centralized inference on policies [Espeholt *et al.*, 2019], by simply adjusting the scheduling policy. Moreover, shared memory and lock-free data structures are used whenever possible, to minimize overhead and save bandwidth. With our distributed RL acceleration techniques, we can successfully train the whole system in less than one day on a single workstation.

3.2.3 RL Agent Design

Complex image-based reinforcement learning is prone to overfitting. A properly designed network architecture for the RL agent is a crucial part of generalization [Cobbe *et al.*, 2019] and computational performance. We design a light-weighted network architecture for both policy and Q-networks (see Figure 1(6)). It consists of residual convolution layers and is shared between actor policies and Q-networks to reduce the total amount of learnable parameters.

In the optimizer node of our distributed RL architecture, we implement both the clipped double Q-learning technique in TD3 [Fujimoto *et al.*, 2018] to reduce the overestimation in off-policy

learning, as well as the maximum entropy RL objective in soft actor-critic (SAC) [Haarnoja *et al.*, 2018] to encourage exploration for improved performance. Specifically, the target Q-value in the Bellman error computation (Eq. (2)) is evaluated using multiple target Q-functions as follows:

$$y = r + \gamma \min_{i=1,2} Q^{\pi}_{\psi'_i}(s', \pi_\phi(s'))$$

And the policy learning objective is revised to maximize both Q-function and entropy of the policy:

$$J_\pi(\phi) = \mathbb{E}_s \mathbb{E}_{a\sim\pi_\phi}[Q^{\pi}_{\psi}(s,a) - \lambda \log \pi_\phi(s)]$$

3.2.4 Sim-to-Real Generalization

As the RL policy learned in a simulator needs to be deployed to unseen real-world scenarios, which induces a large domain transfer gap, we focus on improving the model generalizability by introducing a domain generalization scheme during training, incorporating domain randomization and data augmentation.

Domain randomization [Tobin *et al.*, 2017] is a high-level data manipulation technique with special regard to the internal physical mechanisms from the perspective of generating images. In this perspective, we randomize the configurations in the simulation environment every epoch, forcing the agent to adapt to environments with diverse properties, and also avoid overfitting on certain configurations. At the sensor level, camera position is randomly chosen from a pre-specified range, to reflect sensor installation errors in the real world. Vehicle physical properties (size, mass, etc.) are also randomly chosen from a pre-determined set to reduce the sensitivity to vehicle dynamics.

It has been shown in past literature [Laskin *et al.*, 2020] that applying data augmentation in RL can greatly improved model transfer performance. We apply data augmentation to agent's visual observations, such as rotating and cropping the observed images, which contributes a lot to improve the data efficiency and model generalization during RL training.

4 Evaluation

In this section, we present the baselines, evaluation metrics information and detailed experiment results quantitatively.

4.1 Baselines

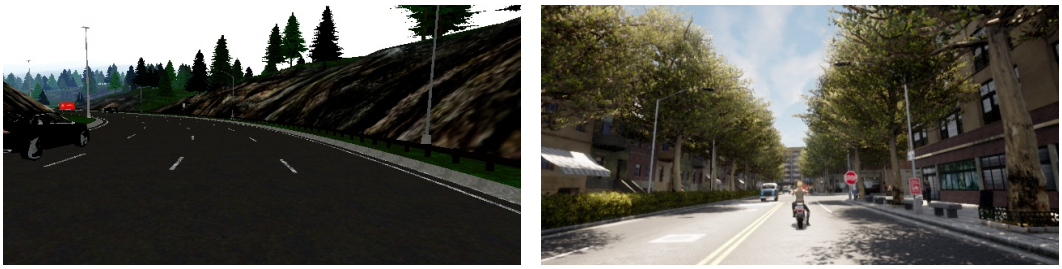
We incorporate five baselines in this study: end-to-end IL [Bojarski *et al.*, 2016], end-to-end RL [Kendall *et al.*, 2019], CycleGAN + IL [Bewley *et al.*, 2019], CycleGAN + RL and the IL variant of our approach (replace the RL policy to the behavior cloning policy). Since CycleGAN is implemented only in testing phase to perform real-to-sim image translation, evaluation results in training phase would keep consistent with fully end-to-end baselines. We re-implement the IL and RL modules of these baselines in our approach for a thorough comparison.

For the training of all IL-based baselines, we follow the treatment described in [Müller *et al.*, 2018]. We collect 1M frames of driving data using a privileged modular pipeline as expert (has access to ground-truth map and obstacle information) provided by CARLA [Dosovitskiy *et al.*, 2017], and add noise to 20% of the expert control outputs to improve the robustness of the learned policy [Codevilla *et al.*, 2018]. For RL-based baselines, all environment and agent configurations are set the same as our proposed approach, and rollout data quantity is set aligned with IL methods for fair comparison.

4.2 Evaluation Metrics

For performance evaluation, we use the following metrics:

- **MPI (m)**: meters per intervention, indicating the level of autonomy of the vehicle, which is widely adopted in real-world autonomous vehicle testing. Interventions are performed when collisions happen or the vehicle doesn't move in more than one minute.
- **SR (%)**: success rate, referring to the proportion of travelled distance from the start to the first intervention with respect to the whole journey in one trial.
- **Std[θ] (°)**: standard deviation of steering angle, reflecting the lateral smoothness of the trajectory.
- **Std[v] (m/s)**: standard deviation of velocity, revealing the longitudinal smoothness of the trajectory.



(a) Training (b) Testing

Figure 3: Examples from the *NoGap* benchmark. Training scenarios use game-like low-quality coarse rendering, while testing scenarios are photo-realistic.

4.3 Evaluation on Semantic Perception Model

The representation model in 3.1 achieves superior accuracy (mIoU) over many popular segmentation models on the BDD100k dataset [Yu *et al.*, 2020], as shown in the results of Table 1. Our model also enjoys greater inference efficiency for more reliable real-time response, capable of running at 31 frames per second on the onboard computer.

Table 1: Perception Model Comparison

Method	Backbone	mIoU (validation)/%	Inference time (ms)
UNet [Ronneberger *et al.*, 2015]	ResNet101	87.2	64
DeepLabv3+ [Chen *et al.*, 2018]	ResNet101	92.5	50
Ours	**EfficientNet-B0**	**93.4**	**32**

4.4 Comparison on Simulation Benchmark

The main challenge of deploying a simulator-trained system to real world is the visual, vehicle dynamics and scenario gap. Previous simulation-based autonomous driving benchmarks, such as *NoCrash* [Codevilla *et al.*, 2019] and CARLA benchmark [Dosovitskiy *et al.*, 2017], do not reflect such challenges effectively. For example, the training and test scenarios are visually similar, and not suited well for comprehensive real-world deployment validation.

To this end, we propose a new *NoGap* benchmark to measure the sim-to-real gaps explicitly and provide better real-world generalization evaluations. In *NoGap* benchmark, the autonomous driving system is trained in “simulator" and then tested in “real-world" (simulator with different configurations). The evaluation settings are described as follows:

- **Visual Gap**: We use different simulator rendering modes to create intentionally introduced visual gaps. During training, low-quality coarse rendering is used, producing game-like images. While in testing, the simulator is switched to photo-realistic rendering, producing images that look like real-world. Please refer to Figure 3 for examples.
- **Dynamics Gap**: Vehicle physical properties are randomly chosen at the beginning of every testing episode, to validate generalization between different vehicle dynamics.
- **Scenario Gap**: 7 CARLA maps are used for training, and 1 reserved for testing, to reflect scenario differences.

We use the MPI metric in simulation evaluations. Inspired by [Codevilla *et al.*, 2019], the environment is reset to move the vehicle to a safe state after an intervention for precise intervention counting. Moreover, background vehicles controlled by built-in modular pipeline in CARLA are generated to emulate real-world traffic and dynamic obstacles.

As shown in Table 2, fully end-to-end solutions fail to generalize from training to testing scenarios due to large visual gaps in RGB inputs. Image-to-image translation method (CycleGAN) can improve testing performance effectively. Furthermore, baselines of training RL with RGB images manifest poorer performance against their IL counterparts by a wide margin, due to the high-dimensional input,

Table 2: Performance validation in simulation, metrics are averaged over 50 runs.

Solutions	Representation	Decision Making	MPI (m)	
			Train	Test
Fully End-to-end IL [Bojarski *et al.*, 2016]	RGB	IL	134.6	16.2
Fully End-to-end RL [Kendall *et al.*, 2019]	RGB	RL	24.9	0.4
End-to-end IL + CycleGAN [Bewley *et al.*, 2019]	Real-to-sim Translation	IL	134.6	46.7
End-to-end RL + CycleGAN [Bewley *et al.*, 2019]	Real-to-sim Translation	RL	24.9	5.3
Ours (IL)	Semantic Representation	IL	306.6	180.9
Ours	Semantic Representation	RL	**449.4**	**332.6**

Table 3: Real world evaluation ($>$ means no intervention over the whole trajectory with specified total length.)

Real World Task			**Solution**							
			Ours (IL)				**Ours**			
Road Topology	Obstacle Setup	Lighting Condition	MPI (m)	SR (%)	Std[θ] ($^\circ$)	Std[v] (m/s)	MPI (m)	SR (%)	Std[θ] ($^\circ$)	Std[v] (m/s)
Straight	✗	Day	92.1	48.9	**1.30**	0.30	**>1163.5**	**100.0**	1.72	**0.19**
	✗	Night	187.1	49.0	**1.67**	**0.18**	**>1304.5**	**100.0**	1.89	0.20
	✓	Day	4.1	16.7	**1.25**	0.37	**34.5**	**75.0**	2.59	**0.30**
Turn	✗	Day	7.2	53.2	**3.03**	0.32	**>214.9**	**100.0**	3.81	**0.23**

while the situation comes to the opposite when training on our segmentation outputs. Thanks to the efficient yet informative representation that eases the burden of RL, our method outperforms all the competing baselines with high generalizability to different maps and manufactured visual gaps.

4.5 Real World Validation

To evaluate the autonomous driving approaches on our test vehicle described in Section B, we categorize the real world tasks based on different attributes: road topology (straight and turn), obstacles setup (with obstacle ahead), and lighting condition (day-time and night-time). We test our approach with the most competitive baseline standing out from simulation evaluation in Section 4.4, the IL version of ours, under the same set of tasks.

To assess the lane-keeping ability on straight and turning roads, we run the vehicle five trials per method for each non-obstacle task, during each of which we set the vehicle to a collision-free position as close as possible to the place where the intervention takes place. The obstacle setup is implemented as the vehicle accelerates from 0m/s and tries to circumvent the obstacle 5m ahead. If the vehicle

Figure 4: The bird's eye views, actual scenarios, raw inputs, and semantic segmentation results (from left to right) for the tasks in Table 3 respectively.

successfully bypasses the obstacle, we record it as a successful trial and reset the vehicle to the initial location for the next trial. Otherwise, it commits a failed trial and counts one more intervention. Table 3 presents the comparisons of real world generalizability between the IL and RL version of our approach on tasks with different attribute settings.

Several observations can be drawn from these experimental results. Our approach achieves much higher MPI and SR on every task, superior to IL version in terms of autonomy and safety. Ours is also generalizable to different lighting conditions, whereas surprisingly, IL version at night-time visibly outperforms itself at day-time. Ours also has smaller velocity std in most of the tasks, while the more conservative IL version has more hold-backs during the real-world tests. Finally, it is observed that ours has better ability to bypass obstacles. This results in larger standard deviation of steering angle of ours than the more conservative IL version, as IL version directly bumps into obstacles and scrapes the curb at turns more often. Abundant qualitative real-world evaluation is performed in Appendix C.

4.6 Computation Performance

Table 4 reports the computational performance of our distributed RL architecture. Due to limited computational resources (a single workstation with 2 RTX3090 GPUs), we use $K = 1$ for all experiments. Compared to non-distributed version which completes the training process in 87.2 hours, our distributed training style ends up with only 12.6 hours, yielding a salient training speedup. With our flexible scheduling scheme, scaling up number of environments N_1 and actors N_2 also yields a consistent speed up. Our semantic perception model is also more light-weighted and effective than CycleGAN, which takes 1/5 computation time to train and also achieves better overall performance.

Table 4: Ablations on Training Efficiency (Unit: hours)

Training Scheme	Representation	RL @2.5M steps	Total
Non-distributed version	5.1	87.2	92.3
CycleGAN, $N_1 = 4, N_2 = 2$	26.9	12.0	38.9
Ours, $N_1 = 4, N_2 = 2$	5.1	12.6	**17.7**
Ours, $N_1 = 4, N_2 = 1$	5.1	13.6	18.7
Ours, $N_1 = 2, N_2 = 1$	5.1	18.7	23.8

5 Conclusion

In this paper, we propose a versatile and efficient RL approach for autonomous driving. By decoupling semantic-meaningful state representation from RL, we alleviate the challenging sim-to-real gap, enhance perception performance, and improve RL decision making abilities. Besides, we tailor a event-driven fully distributed training framework (*OneRL*) for off-policy RL, making it possible to train the whole system in less than one day on a single workstation. We validate the performance and generalization of our approach on a new simulation benchmark (*NoGap*) with intentionally introduced visual,dynamics and scenario gaps. Finally, we build an autonomous vehicle to deploy our approach for real-world evaluation. Our approach could generalize to diverse and unseen complex real-world scenarios and also achieve superior performance compared with various IL and RL baselines.

References

Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir R. Zamir. On Evaluation of Embodied Navigation Agents. pages 1–11, 2018. 1, 3

Mayank Bansal, Alex Krizhevsky, and Abhijit Ogale. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. *arXiv preprint arXiv:1812.03079*, 2018. 3

Aseem Behl, Kashyap Chitta, Aditya Prakash, Eshed Ohn-Bar, and Andreas Geiger. Label efficient visual abstractions for autonomous driving. In *IEEE International Conference on Intelligent Robots and Systems*, pages 2338–2345, 2020. 3

Alex Bewley, Jessica Rigley, Yuxuan Liu, Jeffrey Hawke, Richard Shen, Vinh Dieu Lam, and Alex Kendall. Learning to drive from simulation without real world labels. *Proceedings - IEEE International Conference on Robotics and Automation*, 2019-May:4818–4824, 2019. 2, 3, 4, 6, 8

Mariusz Bojarski, D. Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, L. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *ArXiv*, abs/1604.07316, 2016. 1, 3, 6, 8

Sergio Casas, Abbas Sadat, and Raquel Urtasun. Mp3: A unified model to map, perceive, predict and plan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14403–14412, 2021. 3

Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE international conference on computer vision*, pages 2722–2730, 2015. 1

Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 4

Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018. 4, 7

Dian Chen, Brady Zhou, Vladlen Koltun, and Philipp Krähenbühl. Learning by cheating. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 66–75. PMLR, 30 Oct–01 Nov 2020. 3

Jianyu Chen, Shengbo Eben Li, and Masayoshi Tomizuka. Interpretable end-to-end urban autonomous driving with latent deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–11, 2021. 2

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. 5

Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4693–4700, 2018. 3, 6

Felipe Codevilla, Eder Santana, Antonio Lopez, and Adrien Gaidon. Exploring the limitations of behavior cloning for autonomous driving. *Proceedings of the IEEE International Conference on Computer Vision*, 2019-October(Cvc):9328–9337, 2019. 3, 7

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 3

ONNX Runtime developers. Onnx runtime. https://onnxruntime.ai/, 2021. 15

Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In Sergey Levine, Vincent Vanhoucke, and Ken Goldberg, editors, *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 1–16. PMLR, 13–15 Nov 2017. 2, 3, 4, 6, 7

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, pages 1407–1416. PMLR, 2018. 5

Lasse Espeholt, Raphaël Marinier, Piotr Stanczyk, Ke Wang, and Marcin Michalski. Seed rl: Scalable and efficient deep-rl with accelerated central inference. *arXiv preprint arXiv:1910.06591*, 2019. 5

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018. 5

P. D. Haan, Dinesh Jayaraman, and Sergey Levine. Causal confusion in imitation learning. In *NeurIPS*, 2019. 3

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1861–1870. PMLR, 10–15 Jul 2018. 6

Jeffrey Hawke, Richard Shen, Corina Gurau, Siddharth Sharma, Daniele Reda, Nikolay Nikolov, Przemyslaw Mazur, Sean Micklethwaite, Nicolas Griffiths, Amar Shah, and Alex Kndall. Urban Driving with Conditional Imitation Learning. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 251–257, 2020. 3

Simon Hecker, Dengxin Dai, Alexander Liniger, Martin Hahner, and Luc Van Gool. Learning accurate and human-like driving using semantic maps and attention. *IEEE International Conference on Intelligent Robots and Systems*, pages 2346–2353, 2020. 1, 3

Dan Horgan, John Quan, David Budden, Gabriel Barth-Maron, Matteo Hessel, Hado Van Hasselt, and David Silver. Distributed prioritized experience replay. *arXiv preprint arXiv:1803.00933*, 2018. 5

Junning Huang, Sirui Xie, Jiankai Sun, Qiurui Ma, Chunxiao Liu, Dahua Lin, and Bolei Zhou. Learning a decision module by imitating driver's control behaviors. In *Proceedings of the Conference on Robot Learning (CoRL) 2020*. 1, 2, 3

Zhi Qing Huang, Zhi Wei Qu, Ji Zhang, Yan Xin Zhang, and Rui Tian. End-to-End Autonomous Driving Decision Based on Deep Reinforcement Learning. *Tien Tzu Hsueh Pao/Acta Electronica Sinica*, 48(9):1711–1719, 2020. 3

Zhiyu Huang, Chen Lv, Yang Xing, and Jingda Wu. Multi-Modal Sensor Fusion-Based Deep Neural Network for End-to-End Autonomous Driving with Scene Understanding. *IEEE Sensors Journal*, 21(10):11781–11790, 2021. 1, 3

Alex Kendall, Jeffrey Hawke, David Janz, Przemyslaw Mazur, Daniele Reda, John-Mark Allen, Vinh-Dieu Lam, Alex Bewley, and Amar Shah. Learning to drive in a day. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8248–8254, 2019. 6, 8

B. Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A.Al Sallab, Senthil Yogamani, and Patrick Perez. Deep Reinforcement Learning for Autonomous Driving: A Survey. *IEEE Transactions on Intelligent Transportation Systems*, (February), 2021. 1, 3

Michael Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. *arXiv preprint arXiv:2004.14990*, 2020. 6

Timothée Lesort, Natalia Díaz-Rodríguez, Jean-Franois Goudou, and David Filliat. State representation learning for control: An overview. *Neural Networks*, 108:379–392, 2018. 2

Jesse Levinson, Jake Askeland, Jan Becker, Jennifer Dolson, David Held, Soeren Kammel, J. Zico Kolter, Dirk Langer, Oliver Pink, Vaughan Pratt, Michael Sokolsky, Ganymed Stanek, David Stavens, Alex Teichman, Moritz Werling, and Sebastian Thrun. Towards fully autonomous driving: Systems and algorithms. In *2011 IEEE Intelligent Vehicles Symposium (IV)*, pages 163–168, 2011. 1, 2

Xiaodan Liang, Tairui Wang, Luona Yang, and Eric Xing. Cirl: Controllable imitative reinforcement learning for vision-based self-driving. In *The European Conference on Computer Vision (ECCV)*, September 2018. 3

Jiageng Mao, Minzhe Niu, Chenhan Jiang, Hanxue Liang, Xiaodan Liang, Yamin Li, Chaoqiang Ye, Wei Zhang, Zhenguo Li, Jie Yu, et al. One million scenes for autonomous driving: Once dataset. *arXiv preprint arXiv:2106.11037*, 2021. 3

Michael Montemerlo, Jan Becker, Suhrid Bhat, Hendrik Dahlkamp, Dmitri Dolgov, Scott Ettinger, Dirk Haehnel, Tim Hilden, Gabe Hoffmann, Burkhard Huhnke, and et al. Junior: The stanford entry in the urban challenge. *Journal of Field Robotics*, 25(9):569–597, 2008. 2

Matthias Müller, Alexey Dosovitskiy, Bernard Ghanem, and Vladlen Koltun. Driving Policy Transfer via Modularity and Abstraction. (CoRL), 2018. 1, 2, 3, 6

Błażej Osiński, Adam Jakubowski, Paweł Ziecina, Piotr Miłoś, Christopher Galias, Silviu Homoceanu, and Henryk Michalewski. Simulation-based reinforcement learning for real-world autonomous driving. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6411–6418. IEEE, 2020. 3

Xinlei Pan, Yurong You, Ziyan Wang, and Cewu Lu. Virtual to real reinforcement learning for autonomous driving. *British Machine Vision Conference 2017, BMVC 2017*, 2017. 3

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-Real Transfer of Robotic Control with Dynamics Randomization. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3803–3810, 2018. 1

Aditya Prakash, A. Behl, Eshed Ohn-Bar, Kashyap Chitta, and Andreas Geiger. Exploring data aggregation in policy learning for vision-based urban autonomous driving. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11760–11770, 2020. 3

Kanishka Rao, Chris Harris, Alex Irpan, Sergey Levine, Julian Ibarz, and Mohi Khansari. Rl-cyclegan: Reinforcement learning aware simulation-to-real. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11154–11163, 2020. 1

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 4, 7

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 627–635, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. 3

Axel Sauer, Nikolay Savinov, and Andreas Geiger. Conditional affordance learning for driving in urban environments. In Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 237–252. PMLR, 29–31 Oct 2018. 1

Oliver Scheel, Luca Bergamini, Maciej Wolczyk, Błażej Osiński, and Peter Ondruska. Urban driver: Learning to drive from real-world demonstrations using policy gradients. In *Conference on Robot Learning*, pages 718–728. PMLR, 2022. 3

Stanford Artificial Intelligence Laboratory et al. Robotic operating system. 14

Chen Sun, Jean M Uwabeza Vianney, and Dongpu Cao. Affordance learning in direct perception for autonomous driving. *arXiv preprint arXiv:1903.08746*, 2019. 1

Richard S Sutton, Andrew G Barto, et al. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998. 3

Ardi Tampuu, Tambet Matiisen, Maksym Semikin, Dmytro Fishman, and Naveed Muhammad. A survey of end-to-end driving: Architectures and training methods. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–21, 2020. 1, 2, 3

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019. 4

Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, and et al. Stanley: The robot that won the darpa grand challenge. *Journal of Field Robotics*, 23(9):661–692, 2006. 2

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. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017. 6

Yunkai Wang, Dongkun Zhang, Jingke Wang, Zexi Chen, Yuehua Li, Yue Wang, and Rong Xiong. Imitation Learning of Hierarchical Driving Model: From Continuous Intention to Continuous Trajectory. *IEEE Robotics and Automation Letters*, 6(2):2477–2484, 2021. 1, 3

Yi Xiao, Felipe Codevilla, Akhil Gurram, Onay Urfalioglu, and Antonio M. Lopez. Multimodal End-to-End Autonomous Driving. *IEEE Transactions on Intelligent Transportation Systems*, (201808390010):1–11, 2020. 2, 3

Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 4, 7

Ekim Yurtsever, Jacob Lambert, Alexander Carballo, and Kazuya Takeda. A Survey of Autonomous Driving: Common Practices and Emerging Technologies. *IEEE Access*, 8:58443–58469, 2020. 2

Éloi Zablocki, Hédi Ben-Younes, Patrick Pérez, and Matthieu Cord. Explainability of vision-based autonomous driving systems: Review and challenges. 2021. 2

Appendix

A Reward Designing

In this section, we elaborate on how reward functions are designed addressing all the aspects denoted in Section 3.2.1. For simplicity, we omit the state-action inputs (s, a) in the reward function and describe each part as follows:

- *Speed control:* Instruct the vehicle with current speed v to drive in the desired speed range $[v_{\min}, v_{\text{target}}]$, where $v_{\min} < v_{\text{target}} < v_{\max}$ always holds. It decays linearly when driving too slow or over-speed. $v_{\min}$ and $v_{\max}$ indicate minimum and maximum possible speed, and v_{target} denotes the expected velocity that addresses both safety concerns and travelling efficiency. We choose $v_{\min} = 5m/s, v_{\text{target}} = 6m/s, v_{\max} = 7m/s$ in the course of training.

$$r_{\text{speed}} = \min\{v/v_{\min}, (v_{\max} - v)/(v_{\max} - v_{\text{target}}), 1\} \tag{3}$$

- *Lane center keeping:* Instruct the vehicle to drive in the center of a lane. d is the current distance from vehicle to lane center and $d_{\max}$ is the maximum in-lane distance. We select $d_{\max}$ to be $3m$ in the training proces.

$$r_{\text{center}} = \textsc{Clip}\left(1 - d/d_{\max}, 0, 1\right) \tag{4}$$

- *Heading direction:* Instruct the vehicle to drive aligned with a lane. α is the heading angle difference between the vehicle and the lane, with the maximum angle $\alpha_{\max}$ that We set it to $30°$.

$$r_{\text{heading}} = \textsc{Clip}\left(1 - \alpha/\alpha_{\max}, 0, 1\right) \tag{5}$$

- *Collision and undesired lane crossing penalty:* Finally, we define the collision and undesired lane crossing penalty r_{penalty} as follows, where $\mathbb{I}(\cdot)$ is the incident indicator:

$$r_{\text{penalty}} = 25 \times \mathbb{I}(\text{collision}) + 12 \times \mathbb{I}(\text{cross solid line}) + 15 \times \mathbb{I}(\text{cross double solid line}) \tag{6}$$

The total reward at timestep t is the product of $r_{\text{speed}}, r_{\text{center}}, r_{\text{heading}}$ minus r_{penalty}, enforcing a soft binary AND logic, expecting the agent to pursue all these goals:

$$r = r_{\text{speed}} \cdot r_{\text{center}} \cdot r_{\text{heading}} - r_{\text{penalty}}$$

B Hardware Setup

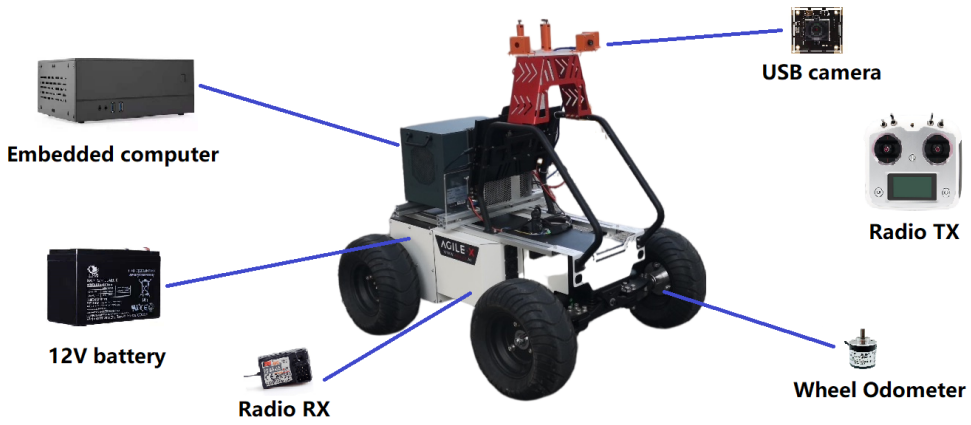


Figure 5: The overall physical system

Our autonomous vehicle is built upon an Agile.X HUNTER Unmanned Ground Vehicle (UGV) with an onboard computer (i7-9700 CPU, 32GB RAM, GeForce RTX 3060 GPU) and a front RGB camera, as shown in Figure 5. The onboard computer runs all the system modules based on Robot Operating System [Stanford Artificial Intelligence Laboratory et al.] and records trajectory statistics by Controller Area Network (CAN) in the real-world validation 4.5. It first applies a low-pass filter on high-level control actions θ, τ from the learned actor network to smooth out noisy signals. These

stable actions are then mapped to the wheel speed ω_1 and steering servo angle ω_2 based on the UGV physical properties. Thereafter, these low-level control commands are delivered to UGV hardware and followed by two PID controllers. To meet the real-time requirement, we utilize quantization and computational graph optimization techniques [developers, 2021] to cut down whole inference duration into 40ms latency, with 100ms control interval.

C Qualitative Real-world experiment

We additionally perform qualitative real-world evaluation on complex, diverse and even unseen scenarios (setup as Figure 6a and 6b), such as being exposed to head-on high beams, traveling on sidewalks, through dense crowds, and even in a garage. Our method can handle lane-following, turning and dynamic obstacle avoidance smoothly, revealing good generalization performance. Please refer to supplementary video for details[6].

(a) Obstacle Avoidance: barrier, bicycle, motorcycle, dummy, person, and chair-like robot (from left to right)

(b) Test in unseen and complex scenarios: high beam, sidewalk, crowds, and garage scenarios (from left to right)

Figure 6: Qualitative real-world experiments

[6] https://www.youtube.com/watch?v=ku8WHoKLwYM

D Hyperparameter Setup

Table 5 lists all the hyperparameters used in the course of training, including both semantic perception and reinforcement learning stages.

Table 5: Hyperparameters in Semantic Perception

Parameter	Value
Semantic Perception	
model	DeepLabv3+
backbone	EfficientNet-B0
optimizer	Adam
learning rate	10^{-3}
batch size	80
loss	Dice loss
training epochs	45
Reinforcement Learning	
algorithm	Soft Actor-Critic
learning rate	$3 * 10^{-4}$
batch size	256
discount (γ)	0.99
target entropy	-2
replay buffer size	10^6
target smoothing factor (τ)	0.02
update to data ratio	0.5