Deductive Reinforcement Learning for Visual Autonomous Urban Driving Navigation

Changxin Huang, *Graduate Student Member, IEEE*, Ronghui Zhang, Meizi Ouyang, Pengxu Wei, Junfan Lin, *Graduate Student Member, IEEE*, Jiang Su, and Liang Lin, *Senior Member, IEEE*

***Abstract*— Existing deep reinforcement learning (RL) are devoted to research applications on video games, e.g., The Open Racing Car Simulator (TORCS) and Atari games. However, it remains under-explored for vision-based autonomous urban driving navigation (VB-AUDN). VB-AUDN requires a sophisticated agent working safely in structured, changing, and unpredictable environments; otherwise, inappropriate operations may lead to irreversible or catastrophic damages. In this work, we propose a deductive RL (DeRL) to address this challenge. A deduction reasoner (DR) is introduced to endow the agent with ability to foresee the future and to promote policy learning. Specifically, DR first predicts future transitions through a parameterized environment model. Then, DR conducts self-assessment at the predicted trajectory to perceive the consequences of current policy resulting in a more reliable decision-making process. Additionally, a semantic encoder module (SEM) is designed to extract compact driving representation from the raw images, which is robust to the changes of the environment. Extensive experimental results demonstrate that DeRL outperforms the state-of-the-art model-free RL approaches on the public CAR Learning to Act (CARLA) benchmark and presents a superior performance on success rate and driving safety for goal-directed navigation.**

***Index Terms*— Autonomous urban driving, deductive reasoning, deep neural networks, reinforcement learning (RL).**

I. Introduction

Vision-based autonomous urban driving navigation (VB-AUDN) is rather under-explored to endow a robotic car/agent with the ability to form long-term driving strategies [1], particularly in changing weather conditions and various complex scenarios [2]–[4]. Considering that inappropriate driving strategies or operations would invite irreversible or disastrous damages, it is desirable to explore how VB-AUDN methods ensure that a car agent could act effectively and safely in structured, changing, and unpredictable environments [5].

Manuscript received July 30, 2020; revised March 23, 2021 and June 15, 2021; accepted August 11, 2021. This work was supported in part by the National Key Research and Development Program of China under Grant 2020AAA0109700 and in part by the National Natural Science Foundation of China under Grant U1811463, Grant 61836012, Grant 61876224, Grant 62006253, Grant U181146, Grant 61836012, and Grant 61976233. *(Corresponding author: Pengxu Wei.)*

Changxin Huang and Ronghui Zhang are with the School of Intelligent Systems Engineering, Sun Yat-sen University, Guangzhou 510275, China (e-mail: huangchx53@mail2.sysu.edu.cn; zhangrh25@mail.sysu.edu.cn).

Meizi Ouyang, Pengxu Wei, Junfan Lin, and Liang Lin are with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510275, China (e-mail: ouymz@mail2.sysu.edu.cn; weipx3@mail.sysu.edu.cn; linjf8@mail2.sysu. edu.cn; linliang@ieee.org).

Jiang Su is with DMAI Great China, Guangzhou 511457, China (e-mail: sujiang@dm-ai.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2021.3109284.

Digital Object Identifier 10.1109/TNNLS.2021.3109284

Current research studies mostly focus on the following approaches: modular pipelines (MPs), imitation learning (IL), and reinforcement learning (RL). MP methods disassemble the VB-AUDN task into a series of easier subproblems, such as scene parsing, path planning, and decision making [6], [7]. Although this strategy makes MPs more interpretable, those MPs heavily depend on intermediate representations, which is hardly applicable for complex scenarios. IL methods aim to mimic human drivers' driving behavior through end-to-end supervised learning [8]. They employ convolutional neural networks (CNNs) to learn the association between original input and control signal based on a mass of driving demonstrations. It requires a huge amount of driving demonstrations to cover the various driving situations which are usually impractical.

Different from those approaches, Deep RL approaches allow the agent to learn the driving policy from exploration data without expert demonstration, which is more suitable for practical applications. Deep RL approaches are rooted in Markov decision process (MDP) [9], which is pioneered by the work of Bellman [10]. RL agents optimize the policy based on the empirical data gathered from their interactions with the environment. It is an exploration-and-exploitation process that reinforces favorable actions while avoiding negative ones that bring the agents low rewards. Existing RL-based VB-AUDN research studies mainly adopt model-free RL (MFRL) to train the agents without explicitly modeling the environment [1], [11], [12]. However, MFRL requires to interact with the environment to obtain millions of data samples to optimize the driving strategy, which is data-inefficient and hinders its applications to complex real-world tasks.

Compared to MFRL, model-based RL (MBRL) approaches are typically more data-efficient [13], [14]. These methods apply an environment model to predict the future and search for favorable outcomes while avoiding the adverse consequences by trial-and-error in realistic environments. Since it is intractable to obtain a ground-truth forward dynamics, approximating a forward dynamics model is regarded as a feasible solution [15]–[18]. Nevertheless, it is challenged by a high-dimensional state space (e.g., pixel-level image), which leads to the difficulty of learning forward dynamics [19], [20]. Due to these approximation errors [18], MBRLs usually fail to achieve competitive performance in comparison with MFRLs.

Essentially, these RL-based approaches optimize driving policy by maximizing long-term cumulative rewards, which may cause the agent to be insensitive to unsafe intermediate situations [21], [22]. It is severely problematic for safety-critical VB-AUDN tasks. Moreover, the scenarios

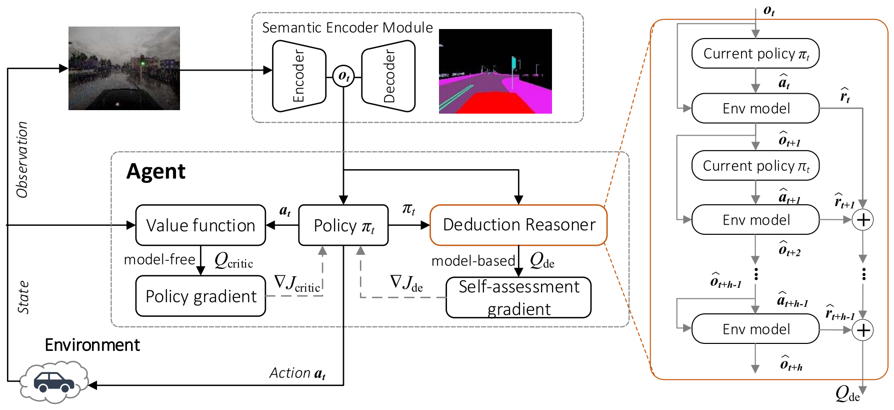


Fig. 1. Overview of our proposed DeRL approach. DeRL consists of a policy network (actor network), a value function (critic network), and DR. The goal of SEM is to learn the driving scenario relevant abstract representations, which is robust to changes in weather and circumstances.

of VB-AUDN are far more complicated than those of many prevalent tasks, e.g., The Open Racing Car Simulator (TORCS) [23] and Atari game [24]. VB-AUDN tasks require the agent to handle the complex visual perception such as the complex surroundings and changing weather conditions [5]. Therefore, it is desirable for RL to deductively reason about the future transitions and reactively navigate with effective responses to unforeseen and complex circumstances.

To address these issues, in this article, we propose deductive RL (DeRL) for the VB-AUDN tasks to mimic deductive human intelligence, which is a combination of model-free and MBRL. We employ deep deterministic policy gradient (DDPG) [25] to update the driving policy in a model-free branch; and propose a deduction reasoner (DR) further improves the driving policy in a model-based branch, as shown in Fig. 1. We utilize the model-based branch to assess the value of the policy. Then the value is used as guidance for policy optimization. Since DR only executes during policy optimization, planning is no longer necessary in the testing stage. To handle the high-dimensional complex scenarios and alleviate the approximation errors, we further propose a semantic encoder module (SEM) to learn low-dimensional driving representations from the raw image observations, which accords with the learning rule (behavior) of the human brain [26], [27]. The compact representation allows the model-based agent to infer in a more robust and efficient way. Overall, we summarize our main contributions as follows.

1) We propose a DeRL approach to promote the performance of VB-AUDN tasks. As far as we know, our proposed DeRL is the first one for vision-based self-driving in a deep RL framework without human demonstration.
2) We introduce a DR to look forward from the current state and to evaluate the value of predicted trajectory to improve the policy learning. DR first predicts the future transitions through a learned environment/dynamics model. It then conducts self-assessment at the predicted trajectory to realize the consequence of current operations, which is combined with the model-free branch to optimize the driving policy.
3) We deploy a novel SEM to learn a low-dimensional intermediate representation from the raw images, providing driving relevant information for driving in urban environments. Our SEM effectively mitigates the approximation errors of the learned environment model and inefficient exploration.
4) Compared with related algorithms without demonstration on the public CAR Learning to Act (CARLA) benchmark [28], the proposed method achieves state-of-the-art performance, demonstrating the superiority of our proposed model for handling a variety of urban driving scenarios including challenging circumstances and weather conditions. Especially, the traffic infraction results show that our agent can drive automatically in a safer way.

The rest of this article is organized as follows. In Section II, we briefly introduce the related research directions to VB-AUDN, including MPs, imitation learning (IL), and RL. Section III introduces the background of RL. The proposed DeRL is elaborated in Section IV. Experiment results and discussions are presented in Section V. Finally, Section VI concludes this article.

II. Related Work

VB-AUDN approaches can be defined as learning a function that maps vision inputs (images) to driving control outputs. Current relevant researches can be approximately divided into three categories: MPs, IL, and RL. We elaborate these three categories for VB-AUDN approaches in the following.

A. MP Based Approaches

MPs decouple the autonomous driving task into multiple subcomponents and solve each subproblem separately. Chen *et al.* [7] split the autonomous driving into two stages, i.e., perception stage for learning driving affordance via CNNs and control stage for mapping affordance to actions according to *ad hoc* driving rules. Dosovitskiy *et al.* [28] improve the perception performance by learning a driving semantic segmentation map and a list of way-points. Though MPs are relatively interpretable, the intermediate submodule divisions rely strictly on human expert analyses, which is extremely rigid for handling changing and unpredictable driving scenarios.

B. IL Based Approaches

IL approaches, pioneered by Pomerleau [29], skip the intermediate step of formulating an explicit driving model and learn the driving skills directly from collected demonstrations. In recent years, the success of deep learning on various computer vision tasks [30]–[34] motivates researchers to adopt deep ConvNets to imitate human driving skills [8], [35]–[38]. However, applying IL directly to a complicated autonomous driving system brings about other challenges. First, it is limited by the requirement of extensive human experiences. For example, Codevilla *et al.* [38] collected more than 400 h of driving demonstrations from CARLA environment [28] using more than 200 GPU-days. Nevertheless, this is impractical in real-world applications. Besides, to create abnormal situations such as collisions and/or going off the road (synthesizing the worst cases), learners are exposed to synthesizing data by adding perturbations to the experts [39]. Notably, the multimodal distributions of data will slow the training process [40]; the human drivers have their driving preferences subjectively and may not even drive in an optimal way for providing reasonable data. Therefore, there is no guarantee that every action of the expert driver is optimal.

C. RL Based Approaches

In order to address the aforementioned issues of MPs and ILs, many research studies attempt to learn autonomous driving policies following an RL framework [41]–[44]. RL system optimizes driving policy in a reasonable way by exploration and interaction with a simulation environment. It encourages favorable movements and punishes improper ones to learn an optimal policy. However, urban driving is more difficult than most previous tasks such as lane following or obstacle avoidance.

Dosovitskiy *et al.* [28] have claimed that the performance of vanilla Asynchronous advantage actor–critic (A3C) [45] is so poor and far from meeting the requirements of practical applications. The main reason is that conventional RL often falls into the local optima due to large action spaces. One effective solution to this dilemma is the learning of better exploration in the context of human demonstrations. Liang *et al.* [12] proposed the controllable imitative RL (CIRL) that uses human demonstrations to initialize the policy networks of DDPG [25], which effectively alleviates the low exploration efficiency for the large continuous action space. However, it relies on human driving data to warm up the RL policy. Another challenge is that MFRL algorithms optimize driving policy by maximizing long-term cumulative rewards, which may cause the agent insensitive to the unsafe intermediate situation [21], [22], which is problematic for autonomous driving systems, as unsafe policies could damage the hardware or invite harm to humans.

Recent works [18], [46]–[48] have demonstrated the effectiveness of combining model-based and model-free algorithms on robotic tasks. For example, Nagabandi *et al.* [18] used the learned environment/dynamics model and the model predictive control (MPC) to gather example trajectories, which are used as "expert" trajectories to train a neural network. Then, the neural network is considered as an initial policy for an MFRL algorithm. This environment model increases the amount of internal simulation for the agent. However, the model-based branch and the model-free branch are trained separately, which is laborious and cumbersome in practice. Pong *et al.* [46] introduced temporal difference models (TDMs), a family of goal-conditioned value functions that can be trained with model-free learning and used for model-based control. However, TDMs can only be used for training goal-conditioned value functions. Racaniére *et al.* [47] and Wang *et al.* [48] construct implicit plans to obtain rollout trajectories by using a learned environment/dynamics model and a model-free policy. The rollout trajectories are then simply encoded into policy networks as additional information. Therefore, the model-based branch only adds additional information but has no clear guidance for policy optimization. In addition, it is time-consuming since they have to do the planning process every step at the testing stage.

The proposed DR is a model-based method, however, different from the aforementioned above work, the rollout trajectory generated from the environment/dynamics model is used for the value estimation, called self-assessment value. The self-assessment value can also optimize the policy network through policy gradient. In this way, when the learned policy is used to predict the action, no planning is required. The usage of the environment/dynamics model in this article is similar to the value prediction network (VPN) [49]. However, VPN relies on discrete states and action spaces and only works in some simple scenarios, i.e., 2-D grid-world and Atari [50]. Our DeRL applies DDPG as the basic RL agent, which can handle continuous action space.

III. Preliminaries

In this section, we will briefly introduce the RL method for VB-AUDN.

A. Notations

Specifically, autonomous driving is formulated as an MDP problem. A standard RL includes an agent interacting with an environment E and receiving a reward r at every time step t in MDP. MDP can be denoted as the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$ is the transition probability function, and $\mathcal{R}$ is the reward function, $\gamma \in (0, 1]$ is the discount factor.

At every time step t, the vehicle agent observes the state $s_t \in \mathcal{S}$, and takes an action a_t according to a policy $\pi : a_t = \pi(s_t)$. Then, the agent transits to the next state $s_{t+1} = \mathcal{P}(s_{t+1}|s_t, a_t)$ and receives a reward $r_t = \mathcal{R}(s_t, a_t)$ as well as a new state $s_{t+1} \in \mathcal{S}$ from the environment. The return from a state s_t as the cumulative γ-discounted reward: $\sum_{k=t}^{T} \gamma^{k-t} \mathcal{R}(s_k, a_k)$. The RL objective is to optimize the policy π by maximizing the expected return from the initial state. According to *Bellman Equation*, the expected return starts from state s_t, takes action a_t, and follows policy π, which is denoted as action-value function $Q_\pi(s_t, a_t)$, i.e., Q-function:

$$Q_\pi(s_t, a_t) = \mathbb{E}_{r_t, s_{t+1} \sim E}\big[r(s_t, a_t) + \gamma\, Q_\pi(s_{t+1}, \pi(s_{t+1}))\big]. \quad (1)$$

B. Deep Deterministic Policy Gradient

DDPG is a representative MFRL method. Considering its remarkable performance for the continuous control problem, DDPG [25] is utilized as our basic model. It consists of a Q-function (the critic) and a policy function (the actor) to learn a deterministic continuous policy. The actor π and the critic Q are learned with deep function optimization, which is parameterized by θ^{π} and θ^{Q}, respectively.

To avoid the agents falling into a local optimum and further improve the exploration efficiency, an Ornstein–Uhlenbeck (OU) noise is introduced to the policy. The output action can be expresses as: $a_t = \pi(s_t) + \mathcal{N}_t$, where $\mathcal{N}_t \in \mathcal{N}_{\text{OU}}$. It utilizes the experience replay strategy which collects the experience tuples (s_t, a_t, r_t, s_{t+1}) stored in a replay buffer B during exploration. In VB-AUDN, these experience tuples are actually driving trajectory tuples. In training stage, the training data are sampled from the replay buffer to optimize the policy. DDPG approximates Q-function by a critic network, which is updated by minimizing the *Bellman* loss

$$L\left(\theta^{Q}\right) = \mathbb{E}_{s_t,a_t,r_t,s_{t+1}\sim B}\left(y_t - Q\left(s_t, a_t|\theta^{Q}\right)\right) \tag{2}$$

where $y_t = r_t + \gamma Q(s_{t+1}, \pi(s_{t+1})|\theta^{Q})$ is the target value. Then the actor learns a Q-optimal driving policy: $\pi(s) = \operatorname{argmax}_a Q(s,a)$ and optimizes the policy parameters by using the sampled policy gradient

$$\begin{aligned}\nabla_{\theta^{\pi}} J \\ \approx \mathbb{E}_{s_t,a_t,r_t,s_{t+1}\sim B}\left[\nabla_a Q\left(s, a|\theta^{Q}\right)\big|_{s=s_t,a=\pi(s_t)} \nabla_{\theta^{\pi}}\pi\left(s|\theta^{\pi}\right)|_{s=s_t}\right].\end{aligned} \tag{3}$$

For stabilize training, DDPG introduced a target actor network $\theta^{\pi'}$ and a target critic network $\theta^{Q'}$, respectively, which are used for calculating the target values. The target network is updated through "soft" update [25]: $\theta' = \tau\theta + (1-\tau)\theta'$, where τ is the update rate.

C. Asymmetric Actor-Critic

Inspired by Pinto *et al.* [51], we revamp the DDPG algorithm into an asymmetric actor–critic structure that uses partial observation o_t to train the actor and exploit access to the full state s_t to train the critic. Thus, the policy comes into: $a_t = \pi(o_t)$. The full-state s_t contains eight measurements obtained from sensors of the CARLA simulator, illustrated in Fig. 2. The observation o_t includes the images (taken by a single monocular forward-facing camera mounted in the center of the roof at the front of the vehicle) and the forward speed v_f. Only the observation o_t is used to produce the control signal in the testing stage.

IV. DeRL for VB-AUDN

In this section, we will first provide an overview of our proposed model for VB-AUDN in Section IV-A. We then introduce an SEM to extract effective feature representation from raw images in Section IV-B. The details of our DeRL are elaborated in Section IV-C. Finally, we describe the reward function in Section IV-D and our DR that contributes to policy learning in Section IV-E.

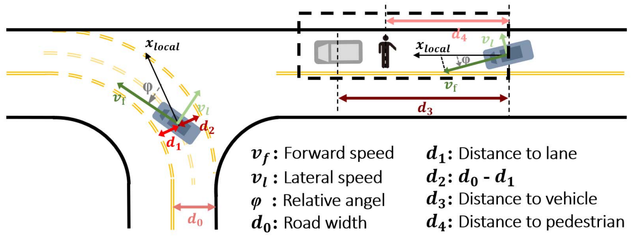


Fig. 2. Illustration of the full state information in DeRL.

A. Overview

To handle the high-dimensional complex scenarios in VB-AUDN, we first propose an SEM to learn a compact representation from the raw image, which is described in Section IV-B. Intuitively, our SEM takes a raw image as input and extracts the driving semantic information into a vector o_t for RL policy π_t. The semantic representation can accelerate the RL training and alleviate the approximation errors of the environment model at the same time.

Our DeRL is built on DDPG [25], which is one of the actor–critic MFRL algorithms. The core of DeRL is that we introduce a deductive reasoner that enables policy to be learned in a model-based manner. Thus, it inherits the advantages of MFRL and MBRL. As shown in Fig. 1, the environment model is regarded as the internal model of the DR. It aims to predict the next state s_{t+1} and reward r_t according to the current state s_t and action a_t. The environment model and reward design are introduced in Sections IV-C and IV-D, respectively. With the environment model and current policy, the following trajectories can be predicted. Meanwhile, DR evaluates the predicted trajectories by summing the estimated rewards, called self-assessment Q_{de}. Finally, the policy learning is guided by both Q_{de} and model-free branch Q_{critic}, which is elaborated in Section IV-E.

B. Semantic Encoder Module

A car agent observes a 2-D image from the environment at each time step. Images could be fed directly into the RL algorithm. However, a raw image generally contains some redundant contents, such as the detailed features of the buildings on the roadside and the color/brightness features brought by the weather. These features are useless for the agent to make driving decisions and even introduce noises. Semantic segmentation extracts the important information in an RGB image, resulting in a more structured and robust representation of a scene. As a result, semantic segmentation is a prevalent option for state representations in many vision-based control tasks [52]–[54]. These approaches use the predicted segmentation (PS) image as the input of RL. However, it is not efficient to deploy an RL algorithm on those high-dimensional images [1]. The motivation of the proposed SEM is more similar to [54]. However, SEM aims to extract a compact representation instead of directly using segmentation images as the input of the RL policy. This allows the RL model to be trained more effectively.

Specifically, our SEM roots in a typical auto-encoder model. Inspired by Xu *et al.* [5], we consider the semantic segmentation prediction as a side task to improve the abstract representation, which performs motion prediction. To further improve the efficiency of the feature representation, the semantic segmentation only retains eight driving task-relevant categories: road, lane-marking, sidewalk, vehicle, pedestrian, pole, traffic sign, and others (background). Intuitively, SEM encodes an observed image into a low-dimensional abstract representation, which is decoded into a semantic segmentation. Our SEM aims to extract low dimension abstract representation with identical distribution from polymorphic distribution images. This encoder process is based on the hypothesis that although the appearance of images varies greatly in different weather conditions or scenes, their semantic segmentations remain consistent. Thus, the learned representations are more robust and insensitive to the variation of weather conditions or scenes.

Variational autoencoder (VAE) [55] can be utilized to extract abstract representation to improve data efficiency in the training process [1], [27]. However, training a vanilla VAE as a standalone model has some limitations. The first limitation is that VAE encodes all information of the observation, which contains task-irrelevant portions. For example, it encodes unimportant detailed grassplots or trees far away from driving roads. The second limitation is that the abstract representation learned by VAE is insensitive to scenario changes. Due to the random sampling of the latent vector, equivalent to adding noise to the encoding, the latent vector encoded by VAE is insensitive to the minor changes of the input, especially the position of perspective. This is also demonstrated in Section V-B.

C. Deduction Reasoner

Our DR employs an environment model to enable the agent to infer the future trajectory, which ensures the agent with cautiousness and reliable decision-making. Specifically, DR first predicts the future transitions through a parameterized environment model. DR then conducts self-assessment at the predicted trajectory to perceive the consequence of current operations, presented as deductive value Q_{de}. Finally, Q_{de} is regarded as a part of the value function to facilitate the driving policy learning.

1) Learning Environment Model: The environment model with the purpose of predicting the next state and estimating the reward that can be obtained. Thus, it contains a transition model T and reward model ξ, as illustrated in Fig. 3. The transition model accepts the observation o_t and action a_t at the time step t to predict the next observation $\hat{o}_{t+1}$. The reward model will estimate the reward of the transition $(o_t,\ a_t,\ o_{t+1})$.

The environment model is trained using the sequence of tuples $(o_t,\ a_t,\ r_t,\ o_{t+1})$ sampled from replay experience. Different from [27] that learns an internal environment model using recurrent neural network (RNN) model, an architecture with three fully-connected layers can perform well in our task. The most important temporary information in driving tasks is the speed of the car. However, speed is a known condition in this task. Therefore, using an RNN may extract the redundant information, while simply fully connected architecture is enough.

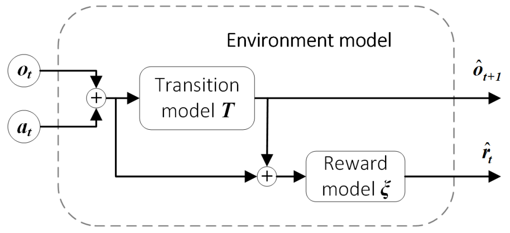


Fig. 3. Environment model accepts the observation o_t and action a_t at time step t to predict the next observation $\hat{o}_{t+1}$ and the reward r_t.

The transition model can be formulated as: $\hat{o}_{t+1} = T(o_t,\ a_t)$, and the loss function is defined as

$$L_T(\theta^T) = ||o_{t+1} - T(o_t, a_t)||^2. \quad (4)$$

The reward model can be formulated as: $\hat{r}_t = \xi(o_t,\ a_t,\ \hat{o}_{t+1})$, and the loss function is defined as

$$L_\xi(\theta^\xi) = ||r_t - \xi(o_t, a_t, o_{t+1})||^2. \quad (5)$$

2) Deduction in Mind: The internal process of deduction module is shown on the right of Fig. 1. We assume that the agent received the observation o_t at time step t, the deduction module will first predict an action $\hat{a}_t$ based on the current policy. Then the environment model will predict the next observation $\hat{o}_{t+1}$ as well as the corresponding reward $\hat{r}_t$. The deduction process will continue with h steps, which denotes the deductive depth. Thus, the deductive process can be formulated as

$$\hat{o}_{t+k} = \begin{cases} T(o_t, \hat{a}_t), & k = 1 \\ T(\hat{o}_{t+k-1}, \hat{a}_{t+k-1}), & 1 < k \leq h. \end{cases} \quad (6)$$

The reward on the predictive trajectory for every time step is formulated as

$$\hat{r}_{t+k-1} = \begin{cases} \xi(o_t, \hat{a}_t, \hat{o}_{t+1}), & k = 1 \\ \xi(\hat{o}_{t+k-1}, \hat{a}_{t+k-1}, \hat{o}_{t+k}), & 1 < k \leq h. \end{cases} \quad (7)$$

Therefore, the self-assessment value is the sum of estimated rewards for each step in the deductive trajectory, which is defined as

$$Q_{\text{de}}(o_t) = \sum_{k=1}^{h} \beta^{k-1} \hat{r}_{t+k-1} + \beta^h V(\hat{o}_{t+h}) \quad (8)$$

where $V(\hat{o}_{t+h})$ is the state value of $\hat{o}_{t+h}$ and β is the discount factor. For MBRL, the performance will drop result from approximated environment model error: when the environment model cannot be learned perfectly, the final policy can highly be suboptimal [18], [46]. Different from goal-reached tasks, autonomous driving demands short-term performance/safety for making a high-efficiency decision/prediction. For this consideration, we set $V(\hat{o}_{t+h}) = 0$ and only accumulate the estimated rewards in the deductive roll-outs to improve the driving policy. Meanwhile, we incorporate a short-term deduction module and only limit the deductive depth to a relatively small number, controlling the cumulative approximation errors without great drift. Finally, Q_{de} will be used as a constraint condition to guide driving policy learning.

D. Designing Reward by Driving Commonsense

We design the reward function to learn driving policy endowed with commonsense like human drivers. The commonsense reward consists of four subrewards with regards to speed, car pose, position and collision. Specifically, the subreward about speed is defined as

$$r_{\text{speed}} = \begin{cases} \upsilon_f(t), & \upsilon_f \leq 20 \\ 40 - \upsilon_f(t), & \upsilon_f \geq 20 \end{cases} \tag{9}$$

where $\upsilon_f(t)$ (in km/h) is the forward speed of the car at time step t. We encourage the agent to drive at a safe and rational speed.

The rotation angle φ reflects the pose of the car which is one of the important indicators for assessing whether the car is driving normally. To urge the car driving along the lane and publish the abnormal steer, the rotation reward $r_{\text{rotation}} = -\upsilon_f(t) \times \tan\varphi$ when the direction command c is go straight, and $r_{\text{rotation}} = -20$ when the car is in the intersection and steer is in opposite direction to c.
Meanwhile, to encourage the car to drive as close as possible to the middle of the road, the position reward is defined as

$$r_{\text{position}} = \begin{cases} 20 \times \left(1 - \dfrac{d}{w}\right), & \text{on} - \text{road} \\ -100, & \text{off} - \text{road} \end{cases} \tag{10}$$

where d is the distance between the center of the agent and the middle of the track, and w is the width of the road. "off-road" represents the vehicle overlaps with the other lane or sidewalk. Finally, the collision reward $r_{\text{collision}}$ is set as -100 for having collision and driving. Thus, the total reward r_{total} is computed as

$$r_{\text{total}} = r_{\text{speed}} + r_{\text{rotation}} + r_{\text{position}} + r_{\text{collision}}. \tag{11}$$

E. DeRL for VB-AUDN

The state-value Q_{critic} from DDPG critic and the self-assessment value Q_{de} from DR are finally weighted and summed to assess the performance of the actor

$$Q(s_t, a_t) = Q_{\text{critic}}(s_t, a_t) + \omega Q_{\text{de}}(o_t) \tag{12}$$

where ω is the weight parameter of Q_{de}. The driving policy is optimized by the policy gradient from two components

$$\nabla_{\theta^\pi} J = \nabla_{\theta^\pi} J_{\text{critic}} + \omega \nabla_{\theta^\pi} J_{\text{de}} \tag{13}$$

where $\nabla_{\theta^\pi} J_{\text{critic}}$ and $\nabla_{\theta^\pi} J_{\text{de}}$ denote the policy gradient calculated from critic of DDPG and DR, respectively. According to (3), model-free policy gradient $\nabla_{\theta^\pi} J_{\text{critic}}$ can be calculated by following equation:

$$\nabla_{\theta^\pi} J_{\text{critic}} = \frac{1}{N} \sum_{i=1}^{N} \nabla_a Q_{\text{critic}}\big(s_i, a = \pi(o_i)|\theta^Q\big) \nabla_{\theta^\pi} \pi(o_i|\theta^\pi). \tag{14}$$

Note that the input of Q_{critic} is the full state s_i, while the input of policy is the partial observation o_i. This is described in Section III-C. The policy gradient from DR can also be obtained by applying the chain rule to the expected return from the start distribution with respect to the actor parameters

$$\nabla_{\theta^\pi} J_{\text{de}} = \frac{1}{N} \sum_{i=1}^{N} \nabla_a Q_{\text{de}}\big(o_i, a = \pi(o_i)|\theta^T, \theta^\xi\big) \nabla_{\theta^\pi} \pi(o_i|\theta^\pi). \tag{15}$$

V. Experiments

To verify the effectiveness of the proposed method, we conduct extensive experiments quantitatively on the public CARLA benchmark, show comprehensive qualitative results and provide further analyses on our proposed DeRL.

The video of testing results shows several navigation examples of our agent for different weather conditions and scenes, which are publicly available.[1]

A. Experiment Setting

1) Simulation Environment: We train and test our car agent on the CARLA simulator [28]. CARLA provides a high fidelity urban driving environment with various traffic situations. It has four tasks defined as follows:

1) *Straight:* Destination is straight ahead of the starting point, and there are no dynamic objects in the environment. The average driving distance to the goal is 200 m in Town01 and 100 m in Town02.
2) *One Turn:* Destination is one turn away from the starting point, and there are no dynamic objects. The average driving distance to the goal is 400 m in Town01 and 170 m in Town02.
3) *Navigation:* No restriction on the location of the destination point relative to the starting point, no dynamic objects. The average driving distance to the goal is 770 m in Town01 and 360 m in Town02.
4) *Navigation With Dynamic Obstacles:* It is the same as the previous three tasks, but with dynamic objects (cars and pedestrians).

For a fair comparison with other state-of-the-art methods, following the same settings, we train the model in Town01 and test it in Town02. Maps of Town01 and Town02 are illustrated in Fig. 4. Weather conditions in the training set contain the clear day, clear sunset, daytime rain, and daytime after rain, while those in the testing set have cloudy daytime and soft rain at sunset that is never used during training stage. Some example images for different weathers are also shown in Fig. 4.

2) Implementation Details:

a) Learning abstract representation: The SEM architecture is shown in Table I, it consists of five convolutional layers of kernel shapes (5, 5, 3, 3, 3) and output channels (32, 64, 128, 256, 64). All strides are set as 2. The output of the convolution stack is then fed into the fully connected layer to produce the low-dimensional latent representation vector with a length of 200. The SEM is trained with image-segmentation pair collected from CARLA under autopilot mode. The dataset contains 110 thousand images and the corresponding ground-truth semantic segmentation. To further

[1]https://1drv.ms/u/s!Aj4kXYOGgb3ua2VurneLrOskNAw?e=Brhqfr

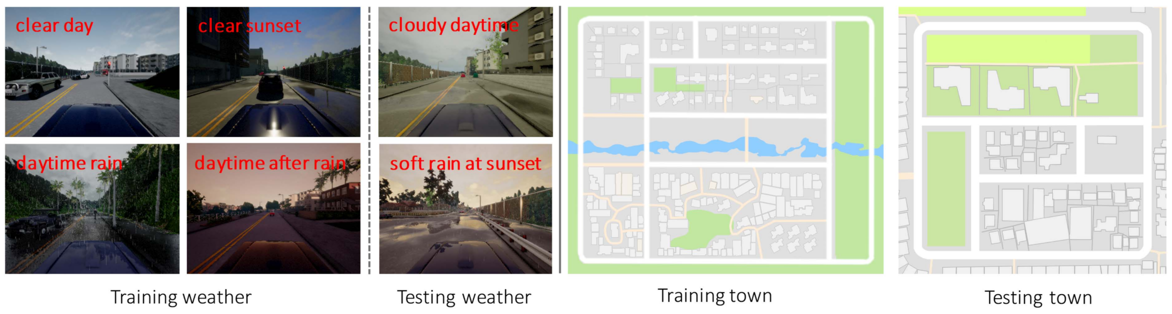


Fig. 4. Example observations of different environment settings. Training conditions contain four different weather in Town 1(Training town) while the rest settings are used for testing.

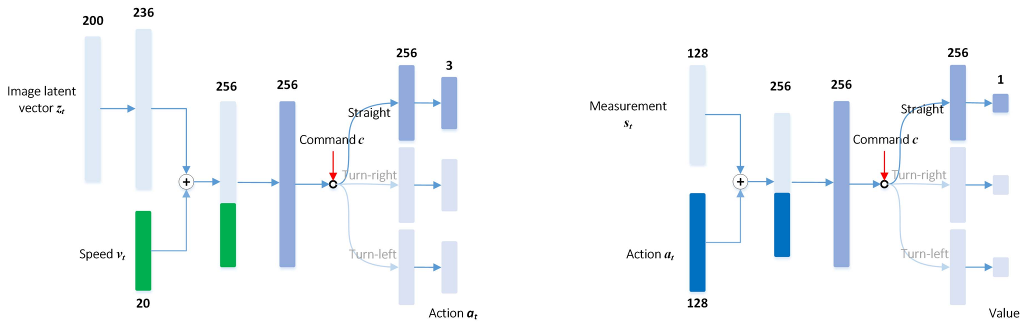


Fig. 5. Left: Actor network architecture of DeRL. The abstract representation encoded by SEM and the car speed are fed into the actor network, and the gating mechanism selectively activates different branches to predict actions. The command control contains Straight, Turn-left and Turn-right. Critic network architecture of DeRL. Right: Critic network architecture of DeRL. The full-state information and action are fed into the critic network to predict the state-action value. The gating mechanism selectively activates different branches to predict value.

TABLE I
CONFIGURATION DETAIL OF SEM

Encoder	Decoder
input image	FC-2304
Conv5-32	Deconv5-256
Conv5-64	Deconv5-128
Conv3-128	Deconv5-64
Conv3-256	Deconv5-32
Conv3-64	Deconv6-16
FC-200	Deconv6-1
latent vector	output segmentation

verify the feasibility of SEM in real-world driving scenario, we simultaneously train the SEM with images and corresponding semantic segmentation predicted by DeepLabv3+ [56], which is pretrained on Cityscapes dataset [57] and fine-tuned with a small number of CARLA images. Despite the simplicity of our SEM, it is demonstrated to effectively mitigate the inefficient exploration of large-scale continuous action for VB-AUDN.

b) Environment model details: Our environment model contains a transition model and a reward model. The transition model only consists of three fully connected layers. The action a_t passes through one fully connected layer of size (3, 128) and is then concatenated with the encoded o_t. The output then passes through two fully connected layers of size (328, 512) and (512, 200) to produce the predicted next observation o_{t+1}.

The reward model contains four fully connected layers. The first layer of size (3, 128) for dealing with a_t. The output then is concatenated with encoded o_t and o_{t+1} and passes through three fully connected layers of size (528, 512), (512, 256), and (256, 1) to predict a reward.

c) Training the RL agent: In this section, we elaborate the detailed network architecture of our asymmetric DDPG. We feed the actor network with observed images and the forward speeds of the car. As shown in Fig. 5, we first concatenate features of the observed images and the forward speeds to provide the observation features. The command c is a categorical variable that selectively activates one of the branches, which is proven to be effective in the urban driving system [8]. The command signal c is provided by the expert, which is similar to previous works like [8], [12]. There are 3 kinds of commands. Straight: drive straight at the next intersection; Turn-left: turn left at the next intersection; Turn-right: turn right at the next intersection. Three policy branches are specifically learned to extract the corresponding hidden knowledge for action prediction. The driving action contains three continuous actions, the steering angle $a_{\text{steer}} \in [-1, 1]$, the acceleration $a_{\text{acceleration}} \in [0, 1]$ and the brake $a_{\text{brake}} \in [0, 1]$, respectively.

In our experiment, the sensor from the CARLA simulator is used to obtain the measurement as the full-state $s_t = (v_f, v_l, \varphi, d_0, d_1, d_2, d_3, d_4)$, whose detailed meaning is shown in Fig. 2. As shown in Fig. 5, the action outputs from the actor network and the full-state information are fed into the critic network to estimate the state-action value. The gating mechanism is also used to selectively activate different branches to predict the state-action value.

We refer to the code from Ben Lau[2] to set the default parameters of DDPG, which is designed for TORCS [23].

[2] *https://yanpanlau.github.io/2016/10/11/Torcs-Keras.html*

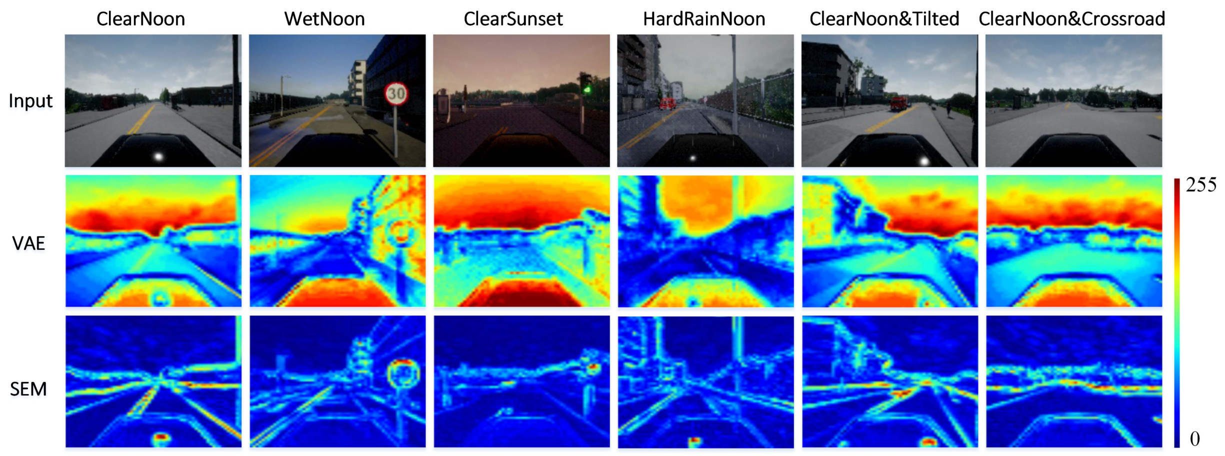
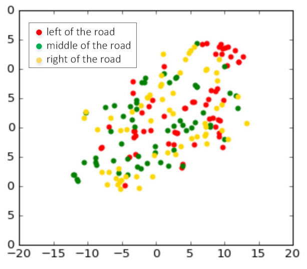
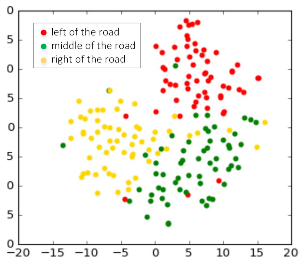


Fig. 6. Comparison of feature maps extracted by VAE and SEM in different scenarios. The first row shows the input image. The second and the third rows indicate the third convolutional layer feature maps extracted by VAE and SEM, respectively.



(a)



(b)

Fig. 7. Distribution of abstract representation z_t extracted by VAE (left) and our SEM (right). To visualize the distribution of abstract representation z_t, we utilize t-SNE to embed high-dimensional z_t into a low-dimensional space of two dimensions. Different colors represent the different locations of the car. The red dot represents the left of the road, the green dot represents the middle of the road and the yellow dot represents the right of the road.

In the training process, the discount factor is set as 0.95. The learning rate of actor network is set as 0.0001 and the learning rate of the critic network is set as 0.001. To ensure the agent training stably, we first warm up the environment model for 20 exploration episodes before policy learning. Both driving tasks and weather are randomly initialized during the training process.

3) Evaluation Metrics: In VB-AUDN task, the objective is to arrive at a destination with as few traffic infractions as possible. Therefore, there are two quantitative evaluation criteria in our experiments, i.e., the success rate and frequency of traffic infractions, which are the same to [28].

The success rate indicates the percentage of successfully completed episodes in goal-directed navigation tasks.

An episode is considered successful when the agent reaches the goal within a time budget; a time budget is the time to reach the destination along the optimal path at a speed of 10 km/h.

The frequency of traffic infraction suggests the average distance traveled between two infractions in "navigation with dynamic obstacles" task. The infraction includes the following five types: driving on the opposite lane, driving on the sidewalk, colliding with other vehicles, colliding with pedestrians, and hitting static objects. The detailed evaluation standards can refer to the protocol of [28].

TABLE II
DISTANCE BETWEEN DIFFERENT DISTRIBUTIONS OF LOCATION

Method	L-M	L-R	M-R
VAE	0.319	0.34	0.161
SEM	0.545	0.58	0.383

B. Learning Abstract Representation

In this section, we will illustrate this viewpoint by visualizing the feature maps and latent vectors extracted by VAE and our SEM, respectively.

The first limitation is that VAE encodes all information of the observation, which contains task-irrelevant portion. We take the third convolutional layer visualization of vae and sem, respectively. As shown in Fig. 6, the features extracted by VAE contain the vehicle body information and a lot of unimportant background details, e.g., the building, the sky or the vegetation. While our SEM can better focus on the lane, the road, and vehicle body information. This part of the information can better reflect the vehicle's position and pose, which is more critical information in driving tasks. What is more the scene background has a low response in SEM feature maps, which can prevent over-fitting to a certain extent.

The second limitation is that the latent vector encoded by VAE is insensitive to the minor changes of the input, especially the position of perspective. In our experiment, we have randomly collected 300 observation images in the simulator and divided them into three categories according to the location of the car, "left of the road," "middle of the road," and "right of the road." To visualize the distribution of abstract representation z_t encoded by VAE, we utilize t-distributed stochastic neighbor embedding (t-SNE) [58] to embed high-dimensional z_t into a low-dimensional space of two dimensions. As illustrated in the left of Fig. 7(a), the distribution of the VAE latent vector is very close regardless of the location of the car. It is dangerous that the agent has no sense of its location on the road. Without the random sampling of the latent vector, our SEM can remain sensitive to the changing of agent location.

TABLE III

QUANTITATIVE EVALUATION OF FOUR TASKS. THE TABLE SHOWS THE PERCENTAGE OF SUCCESSFULLY COMPLETED EPISODES PER TASK DURING DIFFERENT CONDITIONS. WE COMPARE OUR DERL WITH OTHER STATE-OF-THE-ART METHODS. "WITH DEMO" MEANS THAT THE DRIVING POLICY SHOULD BE TRAINED WITH EXPERT DEMONSTRATIONS. "WITHOUT DEMO" MEANS THAT THE DRIVING POLICY DOES NOT REQUIRE THE USE OF EXPERT DEMONSTRATIONS FOR TRAINING. BOLD NUMBER IS THE BEST IN EACH CONDITION

Task	Training condition					New weather					New town					New town and new weather				
	with demo		without demo			with demo		without demo			with demo		without demo			with demo		without demo		
	CIL	CIRL	MP	A3C	**DeRL**	CIL	CIRL	MP	A3C	**DeRL**	CIL	CIRL	MP	A3C	**DeRL**	CIL	CIRL	MP	A3C	**DeRL**
straight	95	98	98	89	**100**	98	100	100	86	**100**	97	**100**	92	74	99	80	98	50	68	**100**
one turn	89	97	82	34	**98**	90	94	95	16	**100**	59	71	61	12	**73**	48	82	50	20	**82**
navigation	86	93	80	14	**93**	84	86	94	2	**96**	40	53	24	3	**62**	44	68	47	6	**68**
dynamic	**83**	82	77	7	75	82	80	**89**	2	82	38	41	24	2	**53**	42	**62**	44	4	60

TABLE IV

INFRACTION ANALYSIS BETWEEN OUR DERL AND OTHER STATE-OF-THE-ART METHODS. THE METRIC OF THE AVERAGE DRIVING DISTANCE BETWEEN TWO INFRACTIONS IS USED. THE HIGHER THE RESULTS, THE BETTER THE AGENT PERFORMS. BOLD NUMBER IS THE BEST IN EACH CONDITION

Task	Training condition				New weather				New town				New town and new weather			
	MP	CIL	A3C	**DeRL**	MP	CIL	A3C	**DeRL**	MP	CIL	A3C	**DeRL**	MP	CIL	A3C	**DeRL**
opposite lane	10.20	**33.40**	0.18	3.80	16.10	**57.30**	0.09	7.80	0.45	**1.12**	0.23	0.49	0.40	0.78	0.21	**0.92**
sidewalk	18.30	12.90	0.75	**30.38**	24.20	**57.00**	0.72	31.21	0.46	**0.76**	0.43	0.46	0.43	**0.81**	0.48	0.62
collision:static	**10.00**	5.38	0.42	8.86	16.10	4.05	0.24	**31.21**	**0.44**	0.40	0.23	0.25	0.45	0.28	0.25	**0.50**
collision:car	**16.40**	3.26	0.58	0.98	**20.20**	1.86	0.85	1.11	0.51	**0.59**	0.41	0.31	**0.47**	0.44	0.37	0.25
collosion:pedestrian	**18.90**	6.35	17.80	10.13	20.40	11.20	20.60	**31.21**	1.40	1.88	2.55	**9.21**	1.46	4.41	2.00	**5.52**

We further measure the distance between two distributions by maximum mean discrepancy (MMD) [59], as shown in Table II. In Table II, L-M denotes the distribution distance between "left of the road" and "middle of the road;" L-R is the distribution distance between "left of the road" and "right of the road;" M-R is the distribution between "middle of the road" and "right of the road." The distance between different distributions of the location of VAE latent is closer than SEM, which illustrates the SEM is more sensitive about location. Meanwhile, as shown in the right of Fig. 7(b), the visualized result further indicates that SEM distinguished different locations of the agent better than VAE.

C. Comparison With State-of-the-Art Methods

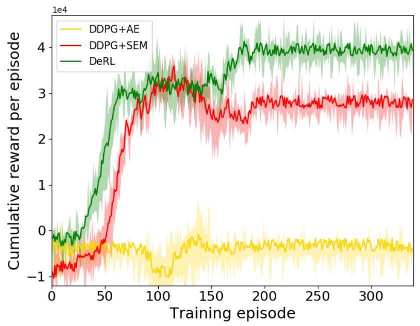
We compare our proposed model with state-of-the-art methods, i.e., MP [28], IL [8], A3C [28] and CIRL [12]. Particularly, both IL and CIRL utilize driver demonstrations in training process.

1) *MP:* We apply the MP method proposed by Dosovitskiy *et al.* [28] that decomposes the driving task among the following subtasks: perception, planning, and continuous control.
2) *IL:* IL serves as an essential tool for learning human skills that are difficult to program by hand. We utilize conditional CIL [8] in our comparison experiment, which uses ConvNets to map an image directly to control.
3) *Asynchronous A3C:* Asynchronous A3C algorithm [45] has demonstrated success in TORCS 3-D car racing simulator [23]. We utilize the released code[3] to evaluate the performance of the A3C algorithm.
4) *CIRL:* The CIRL [12] algorithm also uses expert demonstrations to train an IL ConvNets, which is applied to initialize the policy network of the DDPG framework. It has been demonstrated efficient for large continuous action space.

1) Success Rate of Goal-Directed Navigation: Table III presents the quantitative comparison results between our DeRL and other state-of-the-art methods, where "with demo" means the driving policy should be trained with expert demonstrations, and "without demo" means that the driving policy does not require the use of expert demonstrations for training. It can be observed that our DeRL agent has a higher success rate than CIL except for dynamic tasks. Our agent can achieve superior performance without expensive expert demonstrations. Meanwhile, our proposal outperforms another demonstration-based method CIRL under the training conditions and the new weather tasks. Although CIRL excels in generalizing to the new town, it heavily relies on demonstrations to initialize the parameters of the policy network.

Our DeRL also outperforms MP which learns abundant intermediate features with well-designed rules. Similar to our DeRL, A3C learns driving policy without expert demonstrations. The input of A3C contains two most recent images observed by the agent and a vector of measurements, which includes the current speed of the car, distance to goal, and damage from collisions. By contrast, our DeRL achieved far better performance by only using s single image and the forward speed as the to the policy network. For example, in the training conditions and new weather condition, our DeRL are 93% and 96% on the success rate in the navigation task, which are much higher than A3C. A3C has the worst performance especially in more complicated tasks such as "one turn" and "navigation." One of the reasons for this problem is that the conventional RL is unstable and brittle for complex

[3] *https://github.com/carla-simulator/reinforcement-learning*



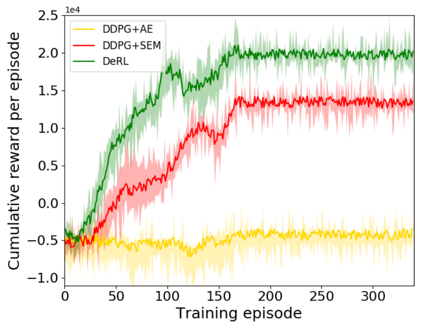


Fig. 8. Effectiveness evaluation of SEM and DR in "straight" task (left) and "one turn" task (right). DDPG+AE is the learning curve of DDPG agent equips with AE encoder. DDPG+ SEM is the learning curve of DDPG agent equips with SEM encoder. DeRL denotes the complete version of the proposed method, which can also be regarded as DDPG equipped with both SEM and DR.

TABLE V
COMPARISON RESULTS BETWEEN AE-DERL AND SEM-DERL IN TERMS OF THE PERCENTAGE OF SUCCESSFULLY COMPLETED EPISODES PER TASK. BOLD NUMBER IS THE BEST IN EACH CONDITION

	Training condition			New weather			New town			New town and weather		
Task	AE	SEM-PS	SEM-GS	AE	SEM-PS	SEM-GS	AE	SEM-PS	SEM-GS	AE	SEM-PS	SEM-GS
straight	72	97	**100**	34	98	**100**	45	76	**99**	7	94	**100**
one turn	55	92	**98**	12	98	**100**	9	32	**73**	3	**94**	82
navigation	31	81	**93**	4	92	**96**	3	21	**62**	1	20	**68**

TABLE VI
ABLATION STUDY ON DR IN TERMS OF THE PERCENTAGE OF SUCCESSFULLY COMPLETED EPISODES PER TASK

	Training condition		New weather		New town		New town and new weather	
Task	DDPG+SEM	DeRL	DDPG+SEM	DeRL	DDPG+SEM	DeRL	DDPG+SEM	DeRL
straight	100	**100**	100	**100**	96	**99**	98	**100**
one turn	89	**98**	94	**100**	50	**73**	46	**82**
navigation	87	**93**	80	**96**	26	**62**	18	**68**
dynamic	**80**	75	80	**82**	7	**53**	16	**60**

real-world tasks [4], which requires the agent to explore for large continuous action space.

2) Infraction Analysis: Table IV shows the average distance (kilometer) driven between two infractions. Compared with A3C, our approach has a longer average distance in almost all tasks. This indicates that our proposed method is safer. Especially, in the task of avoiding driving out of the road and colliding statics, the agent can outperform A3C more than 10 times. In most cases, our DeRL agent achieves the best performance for avoiding pedestrians, which is one of the difficult tasks due to their small size and irregular movements for pedestrians. Moreover, our agent also achieves quite good results under "new town and new weather" conditions, which indicates that the driving policy learned by DeRL is able to generalize to unseen scenarios.

D. Ablation Studies

In this section, we conduct ablation studies to investigate the effectiveness of different components of our model. Specifically, we conduct comparative experiments to analyze SEM and DR in our DeRL. In the following experiments, DDPG is utilized as our base model.

1) *DDPG+AE:* DDPG [25] is an advanced actor-critic MFRL algorithm, which is considered as the base framework of DeRL. DDPG+AE indicates that the observed image is encoded by autoencoder (AE) and then fed to the DDPG.
2) *DDPG+SEM:* We apply SEM to encode the observed image instead of AE. The experimental settings and parameter settings are consistent with DDPG.

1) SEM: Table V reflects the effectiveness of the autoencoder (AE) and our SEM. SEM-PS and SEM-ground-truth segmentation (GS) denote that SEM is trained with predicted and GS maps, respectively. As baseline, AE achieves acceptable results under training conditions. However, its performance declines sharply in all testing conditions. On the contrary, SEM-PS and SEM-GS still perform well in testing conditions. It demonstrates that the features extracted by SEM are more robust and insensitive to the variation of weather conditions or scenes. As shown in Fig. 8, DDPG trained with SEM significantly outperforms AE, indicating that our SEM is able to learn effective driving feature representations.

2) Deduction Reasoner: Tables VI and VII illustrate the importance of DR in terms of success rate and driving safety, respectively. Obviously, our DR substantially promotes the success rate in all tasks, especially in "new town" and "new town and new weather" scenarios. It indicates that DR promotes generalization ability, and the agent can handle changing and unpredictable environments. In addition, one can observe that safety performance has also been improved in most task

TABLE VII

ABLATION STUDY ON DR IN TERMS OF DRIVING SAFETY. BOLD NUMBER IS THE BEST IN EACH CONDITION

Task	Training condition		New weather		New town		New town and new weather	
	DDPG+SEM	DeRL	DDPG+SEM	DeRL	DDPG+SEM	DeRL	DDPG+SEM	DeRL
opposite lane	7.68	**8.62**	5.15	**7.80**	0.22	**0.49**	0.78	**0.92**
sidewalk	4.10	**30.38**	2.37	**31.21**	0.21	**0.46**	0.36	**0.62**
collision:static	2.19	**8.68**	3.43	**31.23**	0.12	**0.25**	0.17	**0.50**
collision:car	**1.25**	0.98	1.40	**1.41**	**0.43**	0.31	0.36	**0.46**
collision:pedestrian	10.05	**10.13**	6.18	**31.21**	2.24	**9.21**	**9.47**	5.52

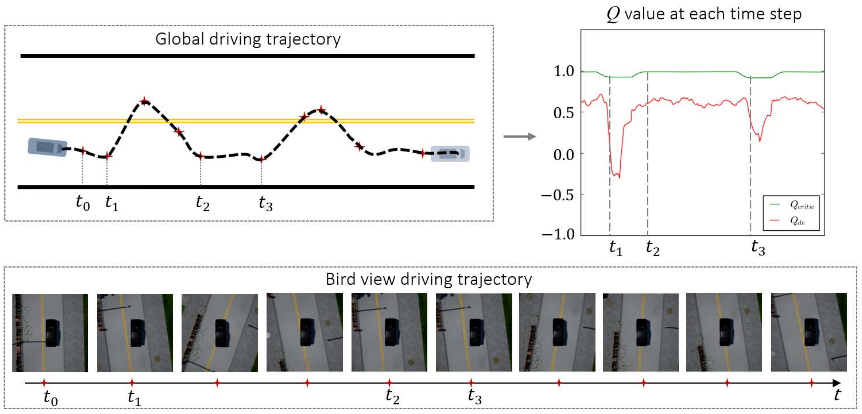


Fig. 9. Utility of Q_{critic} and Q_{de}. Top left: the global trajectory of the agent. Top right: the value of Q_{critic} and Q_{de} that reflect the performance of the agent in the observed time frame. Bottom: the sample frames of the bird view driving trajectory, and each frame corresponds to each time step marked in red in the middle figure. There is a significant decrease in Q_{de} value before the agent drives to the opposite lane while the Q_{critic} value is relatively insensitive. It indicates that Q_{de} is more sensitive to future conditions.

scenarios. This improvement is even more significant in the new weather. In order to show the utility of the self-assessment value provided by DR, we compare the DR value (Q_{de}) with the critic value (Q_{critic}) when the agent tends to drive to the opposite lane, as shown in Fig. 9. At time step t_1, we impose a disturbance on the steer, and the agent tends to drive to the opposite lane, as shown in the top left of Fig. 9. We then observe the DR value (Q_{de}) and the model-free critic value (Q_{critic}), as shown in the top right of Fig. 9. There is a significant decrease in Q_{de} value before the agent drives to the opposite lane while the Q_{critic} value changes rather slightly. This indicates that Q_{de} is far more sensitive to dangerous situations (i.e., driving on the opposite lane) than Q_{critic}, which is beneficial to constrain driving policy learning.

Fig. 8 shows the training process of DDPG+AE, DDPG+SEM, and DeRL in "straight" task and "one turn" task. We compare DDPG+SEM and DeRL to evaluate the effect of DR. In the easiest "straight" task, the agent with DR and without DR achieves comparable results. While in a more complex task scenario, i.e., "one turn" task, DR significantly improves efficiency and increases cumulative rewards, which corresponds to the results of success rate in Table VI.

We further explore how the depth of deduction steps affects performance. As shown in Fig. 10, we compare the performance of different deduction steps: 1 step, 5 steps, 10 steps, and 15 steps. The deduction with 10 steps achieves the best performance in our experiment, while those of 5 steps and 15 steps get the second and third performances, respectively. We believe that a deeper deduction can make DeRL agent perform better, e.g., $\text{DR}_{1-\text{step}} < \text{DR}_{5-\text{step}} < \text{DR}_{10-\text{step}}$.

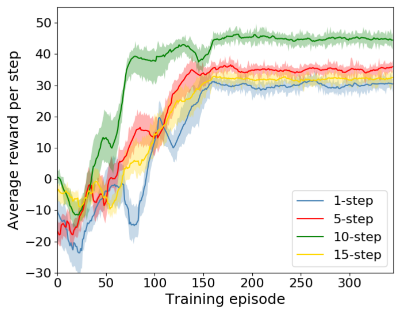


Fig. 10. Evaluation on the deduction depth. h-step indicates that DR deduces the future trajectory with h steps.

However, the performance decreases when the depth of the deduction increases to 15 steps. The main reason is that the approximation errors will increase as the depth of the deduction increases, which inevitably leads to the model performance degradation.

VI. CONCLUSION

In this article, we propose a DeRL method, named DeRL, to address the challenging problems of vision-based autonomous urban driving. Our DeRL embeds a DR into DDPG to resolve the sample inefficiency problem that is intractable in RL research. With the help of DR, our DeRL agent is able to foresee future trajectories from the current state. It also has an appealing property for self-assessing about future actions and states and reactively navigating for an effective response to unforeseen and complex circumstances. In addition, we deploy a novel SEM to learn effective intermediate representations from raw images observed in various driving scenes. Extensive experimental results demonstrate that our DeRL improves the performance of DDPG on vision-based autonomous urban driving tasks both in terms of the success rate and traffic infraction metric. Finally, inspired by Ye *et al.* [60], we plan to equip our DR with meta RL (MRL) to transfer driving policy from simulation to the real world.

References

[1] A. Kendall *et al.*, "Learning to drive in a day," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 8248–8254.

[2] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*. [Online]. Available: http://arxiv.org/abs/1707.06347

[3] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. A. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 387–395.

[4] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1238–1274, 2013.

[5] H. Xu, Y. Gao, F. Yu, and T. Darrell, "End-to-end learning of driving models from large-scale video datasets," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2174–2182.

[6] J. Ziegler *et al.*, "Making Bertha drive—An autonomous journey on a historic route," *IEEE Intell. Transp. Syst. Mag.*, vol. 6, no. 2, pp. 8–20, Apr. 2014.

[7] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "DeepDriving: Learning affordance for direct perception in autonomous driving," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 2722–2730.

[8] F. Codevilla, M. Muller, A. Lopez, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 1–9.

[9] C. C. White and D. J. White, "Markov decision processes," *Eur. J. Oper. Res.*, vol. 39, no. 1, pp. 1–16, 1989.

[10] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[11] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "Safe, multi-agent, reinforcement learning for autonomous driving," 2016, *arXiv:1610.03295*. [Online]. Available: http://arxiv.org/abs/1610.03295

[12] X. Liang, T. Wang, L. Yang, and E. Xing, "CIRL: Controllable imitative reinforcement learning for vision-based self-driving," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2018, pp. 584–599.

[13] K. Asadi, "Strengths, weaknesses, and combinations of model-based and model-free reinforcement learning," M.S. thesis, Dept. Comput. Sci., Univ. Alberta, Edmonton, AB, Canada, 2015. [Online]. Available: https://era.library.ualberta.ca/items/5f163754-3a20-47a0-8649-14d1fd816671, doi: 10.7939/R3C24QW38.

[14] M. Thabet, M. Patacchiola, and A. Cangelosi, "Sample-efficient deep reinforcement learning with imaginary rollouts for human-robot interaction," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 5079–5085.

[15] K. Chua, R. Calandra, R. McAllister, and S. Levine, "Deep reinforcement learning in a handful of trials using probabilistic dynamics models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 4754–4765.

[16] M. Deisenroth and C. E. Rasmussen, "PILCO: A model-based and data-efficient approach to policy search," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 465–472.

[17] T. Kurutach, I. Clavera, Y. Duan, A. Tamar, and P. Abbeel, "Model-ensemble trust-region policy optimization," 2018, *arXiv:1802.10592*. [Online]. Available: http://arxiv.org/abs/1802.10592

[18] A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine, "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 7559–7566.

[19] S. Chiappa, S. Racaniere, D. Wierstra, and S. Mohamed, "Recurrent environment simulators," 2017, *arXiv:1704.02254*. [Online]. Available: http://arxiv.org/abs/1704.02254

[20] C. Finn, I. Goodfellow, and S. Levine, "Unsupervised learning for physical interaction through video prediction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 64–72.

[21] F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 908–918.

[22] J. García and F. Fernández, "A comprehensive survey on safe reinforcement learning," *J. Mach. Learn. Res.*, vol. 16, no. 1, pp. 1437–1480, 2015.

[23] B. Wymann, E. Espié, C. Guionneau, C. Dimitrakakis, R. Coulom, and A. Sumner. (2000). *Torcs, the Open Racing Car Simulator*. Software. [Online]. Available: http://torcs.sourceforge.net

[24] L. Kaiser *et al.*, "Model-based reinforcement learning for Atari," 2019, *arXiv:1903.00374*. [Online]. Available: http://arxiv.org/abs/1903.00374

[25] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," in *Proc. Int. Conf. Learn. Represent.*, 2016, pp. 1–14.

[26] L. Chang and D. Y. Tsao, "The code for facial identity in the primate brain," *Cell*, vol. 169, no. 6, pp. 1013–1028, 2017.

[27] D. Ha and J. Schmidhuber, "Recurrent world models facilitate policy evolution," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 2450–2462.

[28] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proc. Annu. Conf. Robot Learn.*, 2017, pp. 1–16.

[29] D. A. Pomerleau, "ALVINN: An autonomous land vehicle in a neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, 1989, pp. 305–313.

[30] W. Ouyang and X. Wang, "Joint deep learning for pedestrian detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2056–2063.

[31] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Comput. Intell. Neurosci.*, vol. 2018, pp. 1–13, Feb. 2018.

[32] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1988–1996.

[33] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 652–660.

[34] T. Chen, L. Lin, L. Liu, X. Luo, and X. Li, "DISC: Deep image saliency computing via progressive representation learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 6, pp. 1135–1149, Jun. 2016.

[35] J. Zhang and K. Cho, "Query-efficient imitation learning for end-to-end autonomous driving," 2016, *arXiv:1605.06450*. [Online]. Available: http://arxiv.org/abs/1605.06450

[36] Y. Li, J. Song, and S. Ermon, "Infogail: Interpretable imitation learning from visual demonstrations," in *Proc. Adv. Neural Inf. Process. Syst.*, I. Guyon *et al.*, Eds. Red Hook, NY, USA: Curran Associates, 2017, pp. 3812–3822.

[37] R. P. Bhattacharyya, D. J. Phillips, B. Wulfe, J. Morton, A. Kuefler, and M. J. Kochenderfer, "Multi-agent imitation learning for driving simulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 1534–1539.

[38] F. Codevilla, E. Santana, A. M. López, and A. Gaidon, "Exploring the limitations of behavior cloning for autonomous driving," 2019, *arXiv:1904.08980*. [Online]. Available: http://arxiv.org/abs/1904.08980

[39] M. Bansal, A. Krizhevsky, and A. S. Ogale, "ChauffeurNet: Learning to drive by imitating the best and synthesizing the worst," 2018, *arXiv:1812.03079*. [Online]. Available: https://arxiv.org/abs/1812.03079

[40] H. Fan, Z. Xia, C. Liu, Y. Chen, and Q. Kong, "An auto-tuning framework for autonomous vehicles," 2018, *arXiv:1808.04913*. [Online]. Available: https://arxiv.org/abs/1808.04913

[41] S.-Y. Oh, J.-H. Lee, and D.-H. Choi, "A new reinforcement learning vehicle control architecture for vision-based road following," *IEEE Trans. Veh. Technol.*, vol. 49, no. 3, pp. 997–1005, May 2000.

[42] A. El Sallab, M. Abdou, E. Perot, and S. Yogamani, "End-to-end deep reinforcement learning for lane keeping assist," 2016, *arXiv:1612.04340*. [Online]. Available: http://arxiv.org/abs/1612.04340

[43] X. Pan, Y. You, Z. Wang, and C. Lu, "Virtual to real reinforcement learning for autonomous driving," 2017, *arXiv:1704.03952*. [Online]. Available: http://arxiv.org/abs/1704.03952

[44] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforcement learning framework for autonomous driving," *Electron. Imag.*, vol. 2017, no. 19, pp. 70–76, 2017.

[45] V. Mnih *et al.*, "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.

[46] V. Pong, S. Gu, M. Dalal, and S. Levine, "Temporal difference models: Model-free deep RL for model-based control," 2018, *arXiv:1802.09081*. [Online]. Available: http://arxiv.org/abs/1802.09081

[47] S. Racanière *et al.*, "Imagination-augmented agents for deep reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5690–5701.

[48] X. Wang, W. Xiong, H. Wang, and W. Yang Wang, "Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2018, pp. 37–53.

[49] J. Oh, S. Singh, and H. Lee, "Value prediction network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6118–6128.

[50] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents," *J. Artif. Intell. Res.*, vol. 47, pp. 253–279, Jun. 2013.

[51] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, "Asymmetric actor critic for image-based robot learning," in *Proc. Robot., Sci. Syst. Conf.*, Pittsburgh, PA, USA, 2018. [Online]. Available: http://www.roboticsproceedings.org/rss14/p08.html, doi: 10.15607/RSS.2018.XIV.008.

[52] D. S. Chaplot, H. Jiang, S. Gupta, and A. Gupta, "Semantic curiosity for active visual learning," in *Proc. Eur. Conf. Comput. Vis.* Berlin, Germany: Springer, 2020, pp. 309–326.

[53] C. Yu *et al.*, "DS-SLAM: A semantic visual SLAM towards dynamic environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 1168–1174.

[54] Z.-W. Hong *et al.*, "Virtual-to-real: Learning to control in visual semantic segmentation," in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, pp. 4912–4920.

[55] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. Int. Conf. Learn. Represent.*, 2014, pp. 1–15.

[56] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 801–818.

[57] M. Cordts *et al.*, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.

[58] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

[59] L. Song, "Learning via Hilbert space embedding of distributions," Ph.D. dissertation, Univ. Sydney, Sydney, NSW, Australia, 2008.

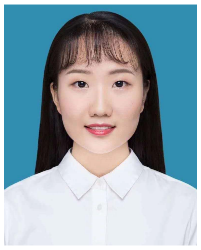
[60] F. Ye, P. Wang, C.-Y. Chan, and J. Zhang, "Meta reinforcement learning-based lane change strategy for autonomous vehicles," 2020, *arXiv:2008.12451*. [Online]. Available: http://arxiv.org/abs/2008.12451

Changxin Huang (Graduate Student Member, IEEE) received the B.S. degree from the School of Automation Science and Engineering, South China University of Technology, Guangzhou, China, in 2015. He is currently pursuing the Ph.D. degree with Sun Yat-sen University, Guangzhou, advised by Prof. Liang Lin.

His current research interests include reinforcement learning and robotics.

Ronghui Zhang received the Ph.D. (Eng.) degree in mechanical and electrical engineering from Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun, China, in 2009.

He is currently an Associate Professor with Sun Yat-sen University, Guangzhou, China. His current research interests include computer vision, intelligent control and intelligent transportation system (ITS).



Meizi Ouyang is currently pursuing the master's degree with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China.

Her research interests include reinforcement learning, deep learning, and computer vision.

Pengxu Wei received the B.S. degree in computer science and technology from China University of Mining and Technology, Beijing, China, in 2011, and the Ph.D. degree from the School of Electronic, Electrical, and Communication Engineering, University of Chinese Academy of Sciences, Beijing, in 2018.

She is currently a Research Scientist with Sun Yat-sen University, Guangzhou, China. Her current research interests include computer vision and machine learning, specifically for data-driven visionand scene image recognition.

Junfan Lin (Graduate Student Member, IEEE) received the B.S. degree in software engineering from Sun Yat-sen University, Guangzhou, China, in 2017, where he is currently pursuing the Ph.D. degree in computer science and technology, advised by Prof. Liang Lin.

His research interests include reinforcement learning and natural language processing.

Jiang Su received the Ph.D. degree from the Department of Electrical and Electronic Engineering, Imperial College London, London, U.K., in 2018.

He joined the Department of Computer Science, University of Cambridge, Cambridge, U.K., as a Research Associate, in 2019. He is currently leading a Research Team with DarkMatter AI Inc., Guangdong, China, focusing on high-performance artificial intelligence (AI) algorithms and robotics research.

Liang Lin (Senior Member, IEEE) is a Full Professor of computer science with Sun Yat-sen University, Guangzhou, China. He served as the Executive Director and a Distinguished Scientist with SenseTime Group, Beijing, China, from 2016 to 2018, leading the Research and Development Teams for cutting-edge technology transferring. He has authored or coauthored more than 200 articles in leading academic journals and conferences [e.g., 20+ articles in IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (TPAMI)/*International Journal of Computer Vision* (IJCV)], and his articles have been cited by more than 16 000 times.

Dr. Lin is a Fellow of IET. He was a recipient of numerous awards and honors including the Wu Wen-Jun Artificial Intelligence Award, the First Prize of China Society of Image and Graphics, the Google Faculty Award in 2012, the Best Paper Diamond Award in IEEE ICME 2017, the Annual Best Paper Award by Pattern Recognition (Elsevier) in 2018, and the ICCV Best Paper Nomination in 2019. His supervised Ph.D. Students received the ACM China Doctoral Dissertation Award, the CCF Best Doctoral Dissertation, and the CAAI Best Doctoral Dissertation. He is an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS and IEEE TRANSACTIONS ON HUMAN-MACHINE SYSTEMS, and served as Area Chairs for numerous conferences such as CVPR, ICCV, SIGKDD, and AAAI.