2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Uncertainty-aware Grounded Action Transformation towards Sim-to-Real Transfer for Traffic Signal Control

Longchao Da, Hao Mei, Romir Sharma and Hua Wei

***Abstract*— Traffic signal control (TSC) is a complex and important task that affects the daily lives of millions of people. Reinforcement Learning (RL) has shown promising results in optimizing traffic signal control, but current RL-based TSC methods are mainly trained in simulation and suffer from the performance gap between simulation and the real world. In this paper, we propose a simulation-to-real-world (sim-to-real) transfer approach called UGAT, which transfers a learned policy trained from a simulated environment to a real-world environment by dynamically transforming actions in the simulation with uncertainty to mitigate the domain gap of transition dynamics. We evaluate our method on a simulated traffic environment and show that it significantly improves the performance of the transferred RL policy in the real world.**

I. Introduction

Traffic Signal Control (TSC) is vital for enhancing traffic flow, alleviating congestion in contemporary transportation systems, and providing widespread societal benefits. It remains an active research area due to the intricate nature of the problem. TSC must cope with dynamic traffic scenarios, necessitating the development of adaptable algorithms to respond to changing conditions.

Recent advances in reinforcement learning (RL) techniques have shown superiority over traditional approaches in TSC [1]. In RL, an agent aims to learn a policy through trial and error by interacting with an environment to maximize the cumulative expected reward over time. The biggest advantage of RL is that it can directly learn how to generate adaptive signal plans by observing the feedback from the environment.

One major issue of applying current RL-based TSC approaches in the real world is that these methods are mostly trained in simulation and suffer from the performance gap between simulation and the real world. While training in simulations offers a cost-effective means to develop RL-based policies, it may not fully capture the complexities of real-world dynamics, limiting RL-based TSC models' practical performance [2]. Simulators often employ static vehicle settings, such as default acceleration and deceleration, whereas real-world conditions introduce substantial variability influenced by factors like weather and vehicle types. These inherent disparities between simulation and reality impede RL-based models trained in simulations from achieving comparable real-world performance, as depicted in Figure 1.

To bridge this gap, prior research has concentrated on enhancing traffic simulators to align more closely with real-world conditions, using real-world data [3]. This enables smoother policy or model transfer from simulation to reality, minimizing performance disparities. Yet, altering internal simulator parameters can be challenging in practice. To tackle this issue, Grounded Action Transformation (GAT) has emerged as a popular technique, aiming to align simulator transitions more closely with reality. However, GAT has predominantly been applied to robotics, with limited exploration in the context of traffic signal control.

In this paper, we present Uncertainty-aware Grounded Action Transformation (UGAT), an approach that bridges the domain gap of transition dynamics by dynamically transforming actions in the simulation with uncertainty. UGAT learns to mitigate the discrepancy between the simulated and real-world dynamics under the framework of grounded action transformation (GAT), which learns an inverse model that can generate an action to ground the next state in the real world with a desired next state predicted by the forward model learned in simulation. Specifically, to avoid enlarging the transition dynamics gap induced by the grounding actions with high uncertainty, UGAT dynamically decides when to transform the actions by quantifying the uncertainty in the forward model. Our experiments demonstrate the existence of the performance gap in traffic signal control problems and further show that UGAT has a good performance in mitigating the gap with higher efficiency and stability.

II. Preliminaries

This section will formalize the traffic signal control (TSC) problem and its RL solutions and introduce the grounded action transformation (GAT) framework for sim-to-real transfer.

A. Concepts of TSC and RL Solutions

In the TSC problem, each traffic signal controller decides the phase of an intersection, which is a set of pre-defined combinations of traffic movements that do not conflict while passing through the intersection. Given the current condition of this intersection, the traffic signal controller will choose a phase for the next time interval Δt to minimize the average queue length on lanes around this intersection. Following existing work [4], [5], [6], an agent is assigned to each traffic signal, and the agent will choose actions to decide the phase in the next Δt. The TSC problem is defined as an MDP which could be characterized by $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, r, \gamma \rangle$

Hua Wei with Longchao Da, Hao Mei, are at School of Computing and Augmented Intelligence (SCAI), Arizona State University, USA, {hua.wei, longchao, hmei7}@asu.edu. Romir Sharma is with the West Windsor-Plainsboro High School South, West Windsor, USA, sharmaromir@gmail.com

The work was partially supported by NSF award #2153311. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

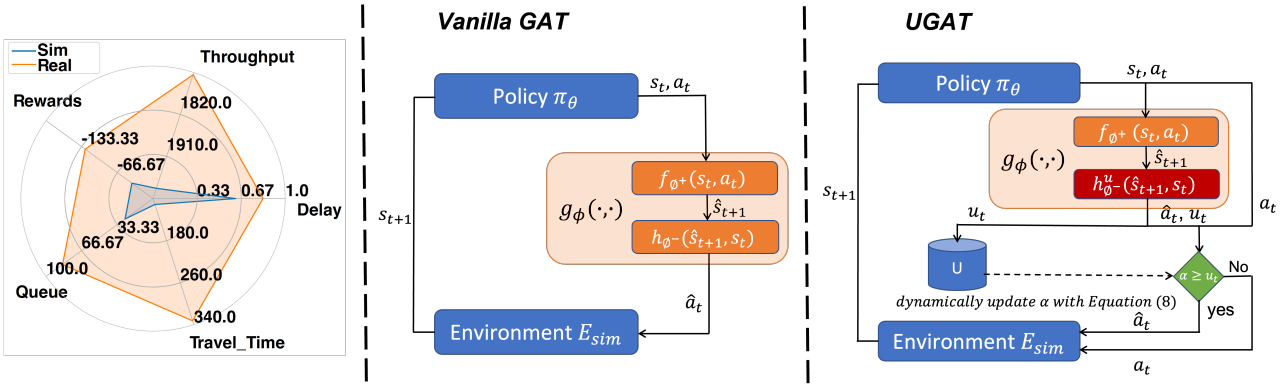


Fig. 1: The performance gap in sim-to-real transfer and the schematic of GAT and UGAT. **Left**: The method [4] trained in simulation has a performance drop when transferred to the real world in all five evaluation metrics in TSC. **Middle**: GAT method takes grounded action $\hat{a}_t$ when a policy returns an action a_t from the E_{sim}. Grounded actions taken with high model uncertainty on $g_{\phi(\cdot,\cdot)}$ will enlarge the transition between P_θ and P^*, making the gap between E_{sim} and E_{real} large and policy learning step not stable. **Right**: UGAT quantifies the model's uncertainty and decide to take or reject the grounded action $\hat{a}_t$ based on the current output of model uncertainty u_t given the current s_t state and action a_t. This behavior will mitigate the gap between P_ϕ and P^* and make the policy learning step stable.

where the state space $\mathcal{S}$ contains each lane's number of vehicles and the current phase of the intersection in the form of the one-hot code, $s_t \in \mathcal{S}$. Action space (discrete) $\mathcal{A}$ is the phase chosen for the next time interval Δt. Transition dynamics $P(s_{t+1}|s_t, a_t)$ maps $\mathcal{S} \times \mathcal{A} \to \mathcal{S}$, describing the probability distribution of next state $s_{t+1} \in \mathcal{S}$. Reward r is an immediate scalar return from the environment calculated as $r_t = -\sum_l w_t^l$, where l is the lane belonging to the intersection and w_t^l is the queue length at each lane. And the γ is the discount factor. Policy π_θ could be represented as the logic of: $\mathcal{S} \to \mathcal{A}$. An RL approach solves this problem by maximizing the long-term expectation of discounted accumulation reward adjusted γ. The discounted accumulated reward is $\mathbb{E}_{(s_t,a_t)\sim(\pi_\theta,\mathcal{M})}[\sum_{t=0}^{T} \gamma^{T-t} r_t(s_t, a_t)]$. Since the action space $\mathcal{A}$ is discrete, we follow the past work using Deep Q-network (DQN) [4] to optimize the π_θ, the above procedure is conducted in simulation environment E_{sim}.

B. Grounded Action Transformation

Grounded action transformation (GAT) is a framework originally proposed in robotics to improve robotic learning by using trajectories from the physical world E_{real} to modify E_{sim}. Under the GAT framework, MDP in E_{sim} is imperfect and modifiable, and it can be parameterized as a transition dynamic $P_\phi(\cdot|s, a)$. Given real-world dataset $\mathcal{D}_{real} = \{\tau^1, \tau^2, \ldots, \tau^I\}$, where $\tau^i = (s_0^i, a_0^i, s_1^i, a_1^i, \ldots, s_{T-1}^i, a_{T-1}^i, s_T^i)$ is a trajectory collected by running a policy π_θ in E_{real}, GAT aims to minimize differences between transition dynamics by finding ϕ^*:

$$\phi^* = \arg\min_\phi \sum_{\tau^i \in \mathcal{D}_{real}} \sum_{t=0}^{T-1} d(P^*(s_{t+1}^i|s_t^i, a_t^i), P_\phi(s_{t+1}^i|s_t^i, a_t^i)) \tag{1}$$

where $d(\cdot)$ is the distance between two dynamics, P^* is the real world transition dynamics, and P_ϕ is the simulation transition dynamics.

To find ϕ efficiently, GAT takes the agent's state s_t and action a_t predicted by policy π_θ as input and outputs a grounded action $\hat{a}_t$. Specifically, it uses an action transformation function parameterized with ϕ:

$$\hat{a}_t = g_\phi(s_t, a_t) = h_{\phi^-}(s_t, f_{\phi^+}(s_t, a_t)) \tag{2}$$

which includes two specific functions: a forward model f_{ϕ^+}, and an inverse model h_{ϕ^-}, as is shown in Fig. 1.

• *The forward model* f_{ϕ^+} is trained with the data from E_{real}, aiming to predict the next possible state $\hat{s}_{t+1}$ given current state s_t and action a_t:

$$\hat{s}_{t+1} = f_{\phi^+}(s_t, a_t) \tag{3}$$

• *The inverse model* h_{ϕ^-} is trained with the data from E_{sim}, aiming to predict the possible action $\hat{a}_t$ that could lead the current state s_t to the given next state. Specifically, the inverse model in GAT takes $\hat{s}_{t+1}$, the output from the forward model, as its input for the next state:

$$\hat{a}_t = h_{\phi^-}(\hat{s}_{t+1}, s_t) \tag{4}$$

Given current state s_t and the action a_t predicted by the policy π_θ, the grounded action $\hat{a}_t$ takes place in E_{sim} will make the resulted s_{t+1} in E_{sim} close to the predicted next state $\hat{s}_{t+1}$ in E_{real}, which makes the dynamics $P_\phi(s_{t+1}|s_t, \hat{a}_t)$ in simulation close to the real-world dynamics $P^*(\hat{s}_{t+1}|s_t, a_t)$. Therefore, the policy π_θ is learned in E_{sim} with P_ϕ close to P^* will have a smaller performance gap when transferred to E_{real} with P^*.

III. Methods

To mitigate the gap in the transition dynamics between traffic simulations and real-world traffic systems, we use the vanilla GAT and analyze its limitations. To overcome it's problem, we propose UGAT to further leverage uncertainty quantification to take grounded action dynamically.

A. Vanilla GAT for TSC

We use the vanilla GAT for the traffic signal control problem by specifying the learning of f_{ϕ^+} and h_{ϕ^-}:

• *The forward model* $f_{\phi^+}(s_t, a_t)$ in traffic signal control problem predicts the next traffic state $\hat{s}_{t+1}$ in the real world given taken action a_t and the current traffic state s_t. We

approximate f_{ϕ^+} with a deep neural network and optimize ϕ^+ by minimizing the Mean Squared Error (MSE) loss:

$$\mathcal{L}(\phi^+) = MSE(\hat{s}^i_{t+1}, s^i_{t+1}) = MSE(f_{\phi^+}(s^i_t, a^i_t), s^i_{t+1}) \tag{5}$$

where s^i_t, a^i_t, s^i_{t+1} are sampled from the trajectories collected from E_{real}.

• *The inverse model* $h_{\phi^-}(\hat{s}_{t+1}, s_t)$ in traffic signal control predicts the grounded action $\hat{a}^i_t$ in simulation E_{sim} to reproduce the same traffic states $\hat{s}_{t+1}$. We approximate h_{ϕ^-} with a deep neural network and optimize ϕ^- by minimizing the Categorical Cross-Entropy (CCE) loss since the target a^i_t is a discrete value:

$$\mathcal{L}(\phi^-) = CCE(\hat{a}^i_t, a^i_t) = CCE(h_{\phi^-}(s^i_{t+1}, s^i_t), a^i_t) \tag{6}$$

where s^i_t, a^i_t, s^i_{t+1} are sampled from the E_{sim} trajectories.

B. Uncertainty-aware GAT

In this section, we will introduce the limitations of the vanilla GAT and propose an uncertainty-aware method on GAT that can benefit from quantifying model uncertainty.

*1) **Model Uncertainty on** g_ϕ:* The vanilla GAT takes supervised learning to train the action transformation function g_ϕ, and grounded action transformation $\hat{a}$ is taken at each step while improving in the E_{sim}. However, the action transformation function g_ϕ could have high model uncertainty on unseen state and action inputs, which is likely to happen during the exploration of RL. With high model uncertainty on g_ϕ, the grounded action $\hat{a}$ in Equation (2) is likely to enlarge the performance gap instead of mitigating it if the high uncertainty action is taken because it will make policy learning unstable and hard to converge.

To overcome the enlarged gap induced by $\hat{a}$ with high model uncertainty in g_ϕ, we need uncertainty quantification methods [7] to keep track of the uncertainty of g_ϕ. Specifically, we would like the action transformation function to output an uncertainty value u_t in addition to $\hat{a}_t$:

$$\hat{a}_t, u_t = g_\phi(s_t, a_t) = h_{\phi^-}(f_{\phi^+}(s_t, a_t), s_t) \tag{7}$$

In general, any methods capable of quantifying the uncertainty of a predicted class from a deep neural network (since h_{ϕ^-} is implemented with deep neural networks) could be utilized, like evidential deep learning (EDL), Concrete Dropout [8], Deep Ensembles [9], etc. In this paper, we explored different state-of-the-art uncertainty quantification methods and found out that they all perform well with our method (their experimental results can be found in Section IV-B.4). We adopted EDL as the default in our method as it performs the best with our method.

Intuitively, during action grounding, whenever model $g_\phi(s_t, a_t)$ returns a grounded action $\hat{a}_t$, if the uncertainty u_t is less than the threshold α, the grounded action $\hat{a}_t$ will be taken in the simulation environment E_{sim} for policy improvement; otherwise, we will reject $\hat{a}_t$ and take the original a_t. This uncertainty quantification allows us to evaluate the reliability of the transformation model and take grounded actions $\hat{a}$ when the model is certain that the resulting transition $P_\phi(s_t, \hat{a}_t)$ would mirror that of the real-world environment E_{real} transition $P^*(s_t, a_t)$. This process enables us to minimize the gap in Equation (2) between the policy training environment E_{sim} and the policy testing environment E_{real}, thereby mitigating the performance gap.

*2) **Dynamic Grounding Rate** α:* The threshold α, which we referred to as the grounding rate, helps us to decide when to filter out $\hat{a}_t$ with uncertainty u_t. One naive approach of deciding the grounding rate α is to treat it as a hyperparameter for training and keep it fixed during the training process. However, since $g_\phi(s_t, a_t)$ keeps being updated during the training process, the model uncertainty of g_ϕ is dynamically changing. Even with the same s_t and a_t, the output u_t and $\hat{a}_t$ from $g_\phi(s_t, a_t)$ could be different in different training iterations.

An alternative yet feasible approach is to set grounding rate α dynamically changing with the model uncertainty during different training iterations. To dynamically adjust the grounding rate with the changing of model uncertainty, we keep track of the model uncertainty u_t of $g_\phi(s_t, a_t)$ during each training iteration. At the end of each iteration i, we update the grounding rate α for the next iteration based on the past record of model uncertainty by calculating the mean

$$\alpha = \frac{\sum_{e=1}^{E} \sum_{t=0}^{T-1} u^e_t}{T \times E} \tag{8}$$

from the logged uncertainties in the last E epochs. This dynamic grounding rate α can synchronously adjust α with the update of g_ϕ and relief efforts on hyper-parameter tuning.

C. Training Algorithm

The overall algorithm for UGAT is shown in Algorithm 1. We begin by pre-training the RL policy π_θ for M epochs in the simulation environment E_{sim}. In each training iteration of UGAT, we collect datasets for both E_{sim} and E_{real}, following the data collection process from [10]. It's worth noting that data collection in E_{real} does not necessarily occur during the training process; it can be obtained from existing offline logged data. With the collected data, we update g_ϕ by training both the forward model f_{ϕ^+} and the inverse model h_{ϕ^-}. Using the updated g_ϕ, we employ policy π_θ to interact with E_{sim} for further policy training. Prior to executing the action a_t generated by $\pi_\theta(s_t)$ in the environment E_{sim}, UGAT grounds the actions using $\hat{a}t$ and the model uncertainty u_t from $g\phi(s_t, a_t)$. If the model uncertainty u_t surpasses the grounding rate α, the grounded action $\hat{a}t$ is rejected, and the original action a_t is executed in the simulation $Esim$. Subsequently, u_t is added to the logged uncertainty U. The RL policy π_θ undergoes updates during the interaction with E_{sim}. After E rounds of intersections, we update α using Equation (8) to prepare for the next round of policy training.

IV. Experiment and Results

In this section, we investigate several aspects of our study: the presence of a performance gap in TSC, the effectiveness of UGAT in mitigating this gap, the influence of dynamic grounding rate α, uncertainty quantification, and action grounding on UGAT' performance, and the stability of UGAT across various uncertainty quantification methods.

Algorithm 1: Algorithm for UGAT with model uncertainty quantification

Input: Initial policy π_θ, forward model f_{ϕ^+}, inverse model h_{ϕ^-}, real-world dataset $\mathcal{D}_{real}$, simulation dataset $\mathcal{D}_{sim}$, grounding rate $\alpha = \inf$

Output: Policy π_θ, f_{ϕ^+}, h_{ϕ^-}

Pre-train policy π_θ for M iterations in E_{sim}
for *i = 1,2, …, I* **do**
  Rollout policy π_θ in E_{sim} and add data to $\mathcal{D}_{sim}$
  Rollout policy π_θ in E_{real} and add data to $\mathcal{D}_{real}$
  # ***Transformation function update step***
  Update f_{ϕ^+} with Equation (5)
  Update h_{ϕ^-} with Equation (6)
  Reset logged uncertainty $U^i = List()$
  # ***Policy training***
  for *e = 1, 2, …, E* **do**
    # ***Action grounding step***
    for *t = 0, 1 ,…, T-1* **do**
      $a_t = \pi(s_t)$
      Calculate $\hat{a}_t$ and u_t with Equation (7)
      if $u_t^e \geq \alpha$ **then**
        $\hat{a}_t = a_t$ # ***Reject grounded action***
      end
      $U.append(u_t^e)$
    end
    # ***Policy update step***
    Improve policy π_θ with reinforcement learning
  end
  Update α with Equation (8)
end

A. *Experiment Settings*

In this section, we introduce the overall environment setup for our experiments, and commonly used metrics.

TABLE I: Real-world Configurations for E_{real}

Setting	accel (m/s^2)	decel (m/s^2)	eDecel (m/s^2)	sDelay (s)	Description
Default	2.60	4.50	9.00	0.00	—
V1	1.00	2.50	6.00	0.50	Lighter loaded vehicles
V2	1.00	2.50	6.00	0.75	Heavier loaded vehicles
V3	0.75	3.50	6.00	0.25	Rainy weather
V4	0.50	1.50	2.00	0.50	Snowy weather

*1) **Environment Setup**:* In this paper, we implement UGAT upon LibSignal [11], an open-sourced traffic signal control library that integrates multiple simulation environments. We treat Cityflow [3] as the simulation environment E_{sim} and SUMO [12] as the real-world environment E_{real}. In later sections, we use E_{sim} and E_{real} by default unless specified. To mimic real-world settings, we consider four configurations in SUMO under two types of real-world scenarios: heavy industry roads and special weather-conditioned roads, with their specific parameters defined in Table I.

• *Default setting* [1]. This is the default parameters for SUMO and CityFlow which describe the normal settings of the vehicle's movement in E_{sim}, with 8 phases TSC strategy.
• *Heavy industry roads.* We model the places where the majority of vehicles could be heavy trucks. In Table I, for the vehicles in $V1$ and $V2$, their accelerating, decelerating, and emergency decelerating rates are more likely to be slower than the default settings. We further consider the vehicles' average startup delay (larger than the default assumption $0s$). As shown in Table I, $V1$ describes roads with lighter-loaded vehicles while $V2$ describes the same roads with heavier-loaded vehicles, and they vary at startup delay.
• *Special weather-conditioned roads.* We examine scenarios with adverse weather conditions, specifically rainy ($V3$) and snowy ($V4$) conditions, as outlined in Table I. In these settings, vehicle acceleration, deceleration, and emergency deceleration rates are reduced compared to the default values, while startup delays are increased. Notably, in snowy weather, the first three rates are lower than in rainy conditions, and the startup delay difference is extended to simulate tire slip.

*2) **Evaluation Metrics**:* Following the literatures in TSC [13], we adopt commonly used traffic signal control metrics as below:
• *Average Travel Time (ATT)* is the average time t it takes for a vehicle to travel through a specific section of a road network. For a control policy, the smaller ATT, the better.
• *Throughput (TP)* is the number of vehicles reached their destinations given amount of time. The larger TP, the better.
• *Reward* is an RL term that measures the return by taking action a_t under state s_t. We use the total number of waiting vehicles as the reward, aligned with Preliminaries. The larger the reward, the fewer waiting vehicles, the better.
• *Queue* is the number of vehicles waiting to pass through a certain intersection in the road network. Smaller is better.
• *Delay* is the average delay per vehicle in seconds and measures the amount of time that a vehicle spends waiting in the network. Smaller is better.

In this work, our goal is to mitigate the performance gap of trained policy π_θ between E_{sim} and E_{real}, we additionally calculate the gap Δ for each referred metric as ATT_Δ, TP_Δ, $Reward_\Delta$, $Queue_\Delta$, and $Delay_\Delta$. For certain metric ψ:

$$\psi_\Delta = \psi_{real} - \psi_{sim} \tag{9}$$

Because in real-world settings, policy π_θ tends to perform worse than in simulation, so the ATT, $Queue$, and $Delay$ in E_{real} are normally larger than those in E_{sim}. Based on the goal of mitigating the gap, improving π_θ performance in E_{sim}, we expect: for ATT_Δ, $Queue_\Delta$, and $Delay_\Delta$, the smaller the better. Because TP_Δ, $Reward_\Delta$ will be negative values, the larger, the better.

B. *Experiment Results*

*1) **Gap between real-world and simulator**:* To investigate the presence of a performance gap in TSC tasks, we conducted an experiment. We employed the Direct-Transfer method,

[1] https://sumo.dlr.de/docs/Definition_of_Vehicles,_Vehicle_Types,_and_Routes.html

TABLE II: The performance using **Direct-Transfer** method compared with using **UGAT** method. The $(\cdot)$ shows the metric gap ψ_Δ from E_{real} to E_{sim} and the $\pm$ shows the standard deviation with 3 runs. The $\uparrow$ means that the higher value for the metric indicates a better performance and $\downarrow$ means that the lower value indicates a better performance.

Setting	Direct Transfer					UGAT				
	$ATT(\Delta\downarrow)$	$TP(\Delta\uparrow)$	$Reward(\Delta\uparrow)$	$Queue(\Delta\downarrow)$	$Delay(\Delta\downarrow)$	$ATT(\Delta\downarrow)$	$TP(\Delta\uparrow)$	$Reward(\Delta\uparrow)$	$Queue(\Delta\downarrow)$	$Delay(\Delta\downarrow)$
V1	158.93(47.69)	1901(-77)	-71.55(-32.11)	47.71(21.59)	0.73(0.11)	**144.72(33.49)**$_{\pm 3.61}$	**1925(-52)**$_{\pm 4.58}$	**-59.38(-19.94)**$_{\pm 3.08}$	**39.58(13.47)**$_{\pm 2.04}$	**0.67(0.05)**$_{\pm 0.01}$
V2	177.27(66.03)	1898(-80)	-87.71(-48.27)	58.59(32.47)	0.76(0.14)	**164.65(53.52)**$_{\pm 12.94}$	**1907(-71)**$_{\pm 13.06}$	**-75.18(-35.74**$_{\pm 8.37}$	**50.25(24.14)**$_{\pm 5.56}$	**0.72(0.10)**$_{\pm 0.01}$
V3	205.86(94.63)	1877(-101)	-101.26(-61.82)	67.62(41.51)	0.76(0.14)	**183.22(71.99)**$_{\pm 13.22}$	**1900(-78)**$_{\pm 13.08}$	**-82.38(-42.94)**$_{\pm 9.11}$	**55.05(28.94)**$_{\pm 6.08}$	**0.72(0.10)**$_{\pm 0.01}$
V4	332.48(221.25)	1735(-252)	-126.71(-87.23)	84.53(58.42)	0.83(0.21)	**284.26(173.03)**$_{\pm 6.67}$	**1794(-184)**$_{\pm 12.05}$	**-111.68(-72.24)**$_{\pm 7.25}$	**74.54(48.43)**$_{\pm 4.82}$	**0.8(0.18)**$_{\pm 0.01}$

training a policy model π_{test} in E_{sim} using the DQN method for 300 epochs. We then directly transferred π_{test} to four distinct E_{real} settings, as detailed in Table I. The results are visualized in Fig. 2, using a radar chart with five metrics indicating performance in two environments, E_{sim} and E_{real}. The blue line connects the metrics in E_{sim}, while the orange line represents E_{real}. A notable performance gap emerges when applying π_{test} to four E_{real} settings compared to E_{sim}. Our experiment confirms the existence of performance gaps, prompting further study into method generalizability.

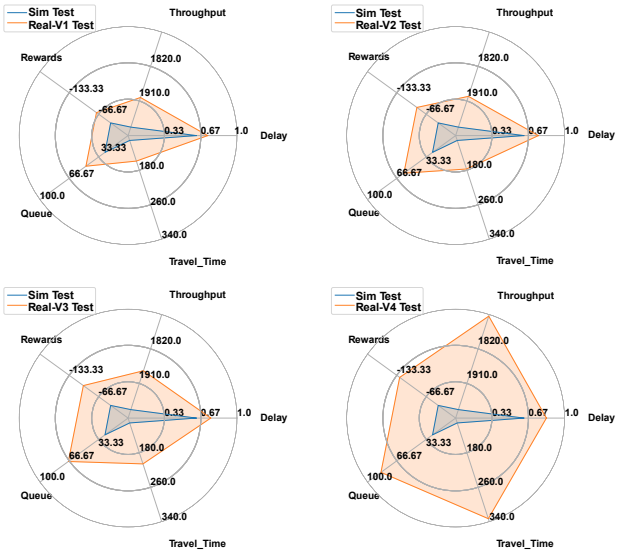


Fig. 2: The performance gap using Direct-Transfer to train in E_{sim} and tested in 4 E_{real} settings.

TABLE III: Direct-Transfer and UGAT performance in E_{sim}

Env	ATT	TP	$Reward$	$Queue$	$Delay$
E_{sim}	**111.23**$_{\pm 0.05}$	**1978**$_{\pm 1}$	**-39.44**$_{\pm 0.03}$	**26.11**$_{\pm 0.05}$	**0.62**$_{\pm 0.01}$

*2) **Gap mitigating by uncertainty-aware UGAT**:* To verify whether the UGAT can effectively mitigate the performance gap, we compare the performance of directly transferring policies trained in E_{sim} to E_{real} with the policies learned under UGAT in four E_{real} settings. Because they are using the same E_{sim}, so performance in E_{sim} eventually converges to stable results with tiny variance as shown in Table III.

In Table II, we use $(\cdot)$ to quantify the performance gap from E_{real} to E_{sim}, as computed with Equation (9). This gap directly reflects the methods' generalization capability from E_{sim} to E_{real}. Our findings are as follows: (1) When comparing UGAT to Direct-Transfer, it effectively reduces the performance gap (Δ). Notably, UGAT exhibits smaller gaps in ATT_Δ, $Queue_\Delta$, and $Delay_\Delta$, while showing larger gaps in TP_Δ and $Reward_\Delta$, indicating effective performance gap mitigation. (2) In terms of the original traffic signal control metrics, UGAT enhances the performance of policy π_ϕ. It achieves lower ATT and higher TP than Direct-Transfer. (3) Table II summarizes experiments across four diverse real-world settings, encompassing five metrics. The results underscore UGAT' robustness and effectiveness in complex environmental conditions.

*3) **Ablation Study**:* In Table IV, We conducted an ablation study on UGAT to assess the impact of its dynamic grounding module, uncertainty quantification module, and action grounding module. We systematically analyzed each module's influence by removing them step by step. When the dynamic grounding module is removed, α is fixed at 0.5. In the third row, when both α and uncertainty are removed, it becomes Vanilla GAT.

TABLE IV: Ablation Study of UGAT on $V1$

$Structure$	ATT_Δ ($\Delta\downarrow$)	TP_Δ ($\Delta\uparrow$)	$Reward_\Delta$ ($\Delta\uparrow$)	$Queue_\Delta$ ($\Delta\downarrow$)	$Delay_\Delta$ ($\Delta\downarrow$)
UGAT	**33.49**$_{\pm 3.61}$	**-52**$_{\pm 4.58}$	**-19.94**$_{\pm 3.08}$	**13.47**$_{\pm 2.04}$	**0.05**$_{\pm 0.01}$
w/o dynamic α	39.12$_{\pm 4.21}$	-72$_{\pm 7.61}$	-25.07$_{\pm 5.71}$	16.88$_{\pm 5.11}$	0.08$_{\pm 0.01}$
w/o α, uncertainty	44.87$_{\pm 4.81}$	-73$_{\pm 12.99}$	-30.59$_{\pm 3.80}$	20.50$_{\pm 1.97}$	0.09$_{\pm 0.01}$
w/o Grounding	47.71$_{\pm 6.73}$	-77$_{\pm 10.64}$	-32.11$_{\pm 4.24}$	21.60$_{\pm 3.12}$	0.11$_{\pm 0.02}$

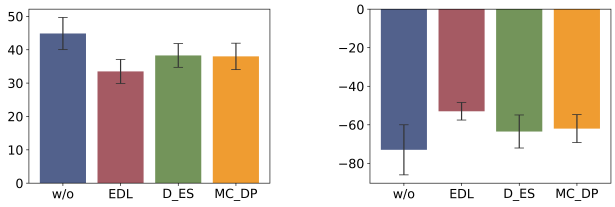
We conducted an ablation study in Table V to investigate the impact of dynamically adjusted grounding rates α on sim-to-real training. We activated the uncertainty quantification module EDL and manually set α values ranging from 0.2 to 0.8. In this study, if the model's uncertainty output u was less than the static α, the action was grounded; otherwise, it was rejected as in Algorithm 1. We compared these results with UGAT, which dynamically adjusts α based on uncertainty. Notably, UGAT significantly improved model performance.

TABLE V: Static vs dynamic α on $V1$

α	ATT_Δ ($\Delta\downarrow$)	TP_Δ ($\Delta\uparrow$)	$Reward_\Delta$ ($\Delta\uparrow$)	$Queue_\Delta$ ($\Delta\downarrow$)	$Delay_\Delta$ ($\Delta\downarrow$)
dynamic	**33.49**$_{\pm 3.61}$	**-52**$_{\pm 4.58}$	**-19.94**$_{\pm 3.08}$	**13.47**$_{\pm 2.04}$	**0.05**$_{\pm 0.01}$
0.2	68.59$_{\pm 7.14}$	-117$_{\pm 12.53}$	-40.42$_{\pm 3.92}$	27.11$_{\pm 4.29}$	0.12$_{\pm 0.05}$
0.4	55.87$_{\pm 7.83}$	-73$_{\pm 13.01}$	-30.69$_{\pm 4.54}$	20.30$_{\pm 3.28}$	0.12$_{\pm 0.01}$
0.5	39.12$_{\pm 4.21}$	-72$_{\pm 7.61}$	-25.07$_{\pm 5.71}$	16.88$_{\pm 5.11}$	0.08$_{\pm 0.01}$
0.6	47.09$_{\pm 2.79}$	-77$_{\pm 4.68}$	-34.11$_{\pm 3.99}$	21.31$_{\pm 2.38}$	0.10$_{\pm 0.03}$
0.8	48.53$_{\pm 6.70}$	-85$_{\pm 9.17}$	-37.85$_{\pm 6.23}$	25.60$_{\pm 2.91}$	0.11$_{\pm 0.01}$

*4) **Different Uncertainty Methods in UGAT**:* In our previous experiments, we utilized EDL [14] for uncertainty quantification. To gain deeper insights into the benefits of

model uncertainty, we conducted additional experiments using various uncertainty quantification methods, namely, EDL, Concrete Dropout [8], and Deep Ensembles [9]. We compared these methods with w/o that excluded the uncertainty module shown in Fig. 3, smaller ATT_Δ is better and larger TP_Δ is better. These experiments demonstrate that model uncertainty quantification methods narrow the performance gap, with EDL outperforming the others, validating its use in our uncertainty quantification module.



(a) ATT_Δ of 4 methods (b) TP_Δ of 4 methods

Fig. 3: Uncertainty investigation across 4 methods on $V1$

V. Related Work

a) Sim-to-real Transfer: The literature on sim-to-real transfer [15] can be generally categorized into three groups. The ***domain randomization*** [16] aims to learn policies that are resilient to changes in the environment. The ***domain adaptation*** [17], [18] tackles the domain distribution shift problem by unifying the source domain and the target domain features that mainly applied in the perception of robots [19], whereas in TSC, the gap is mainly from the dynamics. The ***grounding methods***, improve the accuracy of the simulator concerning the real world by correcting simulator bias. Grounded Action Transformation [10] induces the dynamics of the simulator to match reality-grounded action, showing promising sim-to-real transfer results in robotics. Inspired by GAT, UGAT with the novelty of leveraging uncertainty quantification to enhance action transformation.

b) Uncertainty Quantification: Effective uncertainty quantification (UQ) is essential in current deep learning methods to grasp model limitations and enhance model acceleration and accuracy. Gaussian Process (GPs) [20] is a non-parametric approach for quantifying uncertainty, while another line of research involves using prior distributions on model parameters to estimate uncertainty during training [21]. Evidential Deep Learning (EDL) [14], MC dropout [8], and Deep Ensembles [9] are representative methods that leverage parametric models. This paper experiments on EDL, MC dropout, and Deep Ensembles to explore their benefits.

VI. Conclusion

In this paper, we identify the performance gap in traffic signal control problems and introduce UGAT, an uncertainty-aware grounding action transformation method, to dynamically adapt actions in simulations. Our experiments confirm that UGAT effectively reduces the performance gap with improved stability. This work represents progress in enhancing the real-world applicability of RL-based traffic signal control models, our code can be found at `https://github.com/DaRL-LibSignal/UGAT.git`.

References

[1] M. Noaeen, A. Naik, L. Goodman, J. Crebo, T. Abrar, Z. S. H. Abad, A. L. Bazzan, and B. Far, "Reinforcement learning in urban network traffic signal control: A systematic literature review," *Expert Systems with Applications*, p. 116830, 2022.

[2] Y. Jiang, T. Zhang, D. Ho, Y. Bai, C. K. Liu, S. Levine, and J. Tan, "Simgan: Hybrid simulator identification for domain adaptation via adversarial reinforcement learning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 2884–2890.

[3] H. Zhang, S. Feng, C. Liu, Y. Ding, Y. Zhu, Z. Zhou, W. Zhang, Y. Yu, H. Jin, and Z. Li, "Cityflow: A multi-agent reinforcement learning environment for large scale city traffic scenario," in *The world wide web conference*, 2019, pp. 3620–3624.

[4] H. Wei, G. Zheng, H. Yao, and Z. Li, "Intellilight: A reinforcement learning approach for intelligent traffic light control," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2496–2505.

[5] C. Chen, H. Wei, N. Xu, G. Zheng, M. Yang, Y. Xiong, K. Xu, and Z. Li, "Toward a thousand lights: Decentralized deep reinforcement learning for large-scale traffic signal control," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 3414–3421.

[6] H. Wei, N. Xu, H. Zhang, G. Zheng, X. Zang, C. Chen, W. Zhang, Y. Zhu, K. Xu, and Z. Li, "Colight: Learning network-level cooperation for traffic signal control," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 1913–1922.

[7] H. D. Kabir, A. Khosravi, M. A. Hosen, and S. Nahavandi, "Neural network-based uncertainty quantification: A survey of methodologies and applications," *IEEE access*, vol. 6, pp. 36 218–36 234, 2018.

[8] Y. Gal, J. Hron, and A. Kendall, "Concrete dropout," *Advances in neural information processing systems*, vol. 30, 2017.

[9] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," *Advances in neural information processing systems*, vol. 30, 2017.

[10] J. Hanna and P. Stone, "Grounded action transformation for robot learning in simulation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[11] H. Mei, X. Lei, L. Da, B. Shi, and H. Wei, "Libsignal: An open library for traffic signal control," *arXiv preprint arXiv:2211.10649*, 2022.

[12] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *2018 21st international conference on intelligent transportation systems (ITSC)*. IEEE, 2018, pp. 2575–2582.

[13] H. Wei, G. Zheng, V. Gayah, and Z. Li, "Recent advances in reinforcement learning for traffic signal control: A survey of models and evaluation," *ACM SIGKDD Explorations Newsletter*, vol. 22, no. 2, pp. 12–18, 2021.

[14] M. Sensoy, L. Kaplan, and M. Kandemir, "Evidential deep learning to quantify classification uncertainty," *Advances in neural information processing systems*, vol. 31, 2018.

[15] W. Zhao, J. P. Queralta, and T. Westerlund, "Sim-to-real transfer in deep reinforcement learning for robotics: a survey," in *2020 IEEE symposium series on computational intelligence (SSCI)*. IEEE, 2020, pp. 737–744.

[16] J. P. Tobin, *Real-World Robotic Perception and Control Using Synthetic Data*. University of California, Berkeley, 2019.

[17] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance. arxiv 2014," *arXiv preprint arXiv:1412.3474*, 2019.

[18] T. Han, C. Liu, W. Yang, and D. Jiang, "Learning transferable features in deep convolutional neural networks for diagnosing unseen machine conditions," *ISA transactions*, vol. 93, pp. 341–353, 2019.

[19] S. James, P. Wohlhart, M. Kalakrishnan, D. Kalashnikov, A. Irpan, J. Ibarz, S. Levine, R. Hadsell, and K. Bousmalis, "Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 627–12 637.

[20] M. Seeger, "Gaussian processes for machine learning," *International journal of neural systems*, vol. 14, no. 02, pp. 69–106, 2004.

[21] Y. Xue, S. Cheng, Y. Li, and L. Tian, "Reliable deep-learning-based phase imaging with uncertainty quantification," *Optica*, vol. 6, no. 5, pp. 618–629, 2019.