Self-Learned Autonomous Driving at Unsignalized Intersections: A Hierarchical Reinforced Learning Approach for Feasible Decision-Making

Mohammad Al-Sharman, *Graduate Student Member, IEEE*, Rowan Dempster, Mohamed A. Daoud, Mahmoud Nasr, Derek Rayside, *Member, IEEE*, and William Melek, *Senior Member, IEEE*

***Abstract*— Reinforcement learning-based techniques, empowered by deep-structured neural nets, have demonstrated superiority over rule-based methods in terms of making high-level behavioral decisions due to qualities related to handling large state spaces. Nonetheless, their training time, sample efficiency and the feasibility of the learnt behaviors remain key concerns. In this paper, we propose a novel hierarchical reinforcement learning-based decision-making architecture for learning left-turn policies at unsignalized intersections with feasibility guarantees. The proposed technique is comprised of two layers; a high-level learning-based behavioral planning layer which adopts soft actor-critic (SAC) principles to learn high-level, non-conservative yet safe, driving behaviors, and a low-level motion planning layer that uses Model Predictive Control (MPC) framework to ensure feasibility of the two-dimensional left-turn maneuver. The high-level layer generates reference signals of velocity and yaw angles for the ego vehicle taking into account safety and collision avoidance with the intersection vehicles, whereas the low-level motion planning layer solves an optimization problem to track these reference commands taking into account several vehicle dynamic constraints and ride comfort. While training the behavioral SAC-based planning layer, We develop an adaptive entropy regularization technique that results in faster convergence, higher mean rewards, and lower collision rates. We validate the proposed decision-making scheme in simulated environments and compare with other model free Reinforcement Learning (RL) baselines. The results demonstrate that the proposed integrated framework possesses better training and navigation capabilities.**

***Index Terms*— Urban autonomous vehicles, unsignalized intersections, decision-making, deep reinforcement learning, model predictive control, soft-actor-critic, left-turn maneuvers.**

Manuscript received 31 August 2022; revised 12 March 2023 and 27 April 2023; accepted 8 June 2023. Date of publication 22 June 2023; date of current version 1 November 2023. This work was supported by Natural Sciences and Engineering Research Council of Canada (NSERC) Collaborative Research and Development (CRD) 537104-18, in partnership with General Motors Canada and the Society of Automotive Engineers (SAE) AutoDrive Challenge. The Associate Editor for this article was C. Wei. *(Corresponding author: Mohammad Al-Sharman.)*

Mohammad Al-Sharman, Rowan Dempster, Mahmoud Nasr, and Derek Rayside are with the Department of Electrical and Computer Engineering, and the Waterloo Autonomous Vehicle Team (WATonomous), SAE AutoDrive Challenge (Watonomous.ca), University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: mkalsharman@uwaterloo.ca; r2dempster@uwaterloo.ca; mmmnasrm@uwaterloo.ca; drayside@uwaterloo.ca).

Mohamed A. Daoud and William Melek are with the Department of Mechanical and Mechatronics Engineering, and the WATonomous, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: mohamed.daoud1@uwaterloo.ca; william.melek@uwaterloo.ca).

Digital Object Identifier 10.1109/TITS.2023.3285440

List of Symbols

S	Observed States.
A	Set of actions.
T	Transition function.
γ	Discount factor.
R	Set of rewards.
v_{lim}	Maximum urban speed limit.
λ	Success rate over 10 consecutive evaluations.
$\mathcal{D}$	Replay buffer.
π_ψ	Policy that maps the states into actions.
ξ	Independent noise sequences.
α	Entropy regularization coefficient.
$\mathbb{R}$	Real numbers.
$\dot{(\cdot)}$	Time derivative.
$\mathbf{x}$	Vehicle model states.
CG	Center of gravity of the vehicle.
δ_f	Front wheel steering angle.
L	Distance between front axis and rear axis of the vehicle.
β_s	Vehicle side-slip angle.
θ	Yaw angle of the vehicle.
l_r	Distance between rear axle and CG.
$\mathbf{u}$	Control actions.
$\mathbf{z}$	Augmented system states.
Q	States error weighting matrix.
R	Change in control actions weighting matrix.
t_0	Initial time.
T_H	Prediction horizon.
ICR	Instantaneous center of rotation.
$(\cdot)_e$	Ego vehicle's state.
$(\cdot)_{tar}$	Target vehicle's state.

I. Introduction

A. Motivation

URBAN autonomous driving, as a mechatronic application, is foreseen to have the potential to enhance both safety and efficiency of transportation in environments with complicated traffic conditions [1], [2], [3], [4]. Unlike autonomous driving on highways, autonomous driving in urban settings is challenging due to complications associated with understanding the environment in which the ego vehicle

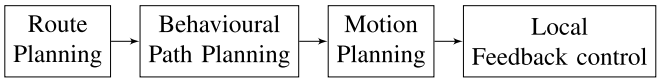


Fig. 1. Decision-making processes in urban autonomous vehicles.

must handle complex driving scenarios where the intentions of other road users along with their interactions are not deterministic [5], [6], [7]. Driving at intersections, in particular, is regarded as critical by the vast majority of human drivers as they must decide on a passing time and carry out crash-free crossing maneuvers. Conducting such maneuvers, at unsignalized intersections, is considered more challenging due to the absence of traffic control signals making the motion inference of other intersection-users highly intractable. The unknown intentions of the intersection participants are unforeseeable as the current states and other hidden variables such as final destinations are unknown [8], [9]. As a consequence, current automated vehicles, even fully autonomous ones, are incapable of always navigating safely and cannot guarantee crash-free maneuvers in such environments due to the limited capability of behavioural planners in handling such large partially-observable environments. Hence, overcoming technical hurdles in the development of more capable, yet safer, urban decision-making techniques is worth investigating.

B. Decision-Making for Urban Autonomous Vehicles

For urban autonomous driving, decision-making processes are defined as a hierarchy of four cascaded layers (see Fig.1); starting with global route planning, followed by the high-level behavioural planning and low-level motion planning layers, and the local feedback control completes the scheme [10], [11]. At the very top layer, given a predetermined destination, the autonomous decision-making system uses route planning algorithms to generate the optimal path using the road network as a network graph. Once the optimal route has been determined, the behavioral path planner is in charge of selecting appropriate driving behaviour based on perceptual observations of the environment. In the context of urban driving at intersections, the behavioural path planner is responsible for selecting a driving behaviour for unsignalized intersection-traversal maneuvers based on the environment conditions, including the unknown states and hidden variables of the intersection users which may generate multiple possible trajectories. Finally, this driving behaviour has to be mapped to a trajectory which will be tracked by a feedback controller considering aspects of the vehicle's dynamic model constraints, the guarantee of ride comfort, and safety [12], [13].

C. Decision-Making at Unsignalized Intersections

Numerous studies have investigated developing safe decision-making and motions planning algorithms at unsignalized intersections [14], [15]. These algorithms have tackled two main issues; inferring the intention of other agents and planning the ego vehicle motion. The intention inference problem is a classification problem where intentions are classified based on the driving behaviour [16]. Index-based and Machine learning-based approaches have been introduced to predict the driver intent at unsignalized intersection [17], [18]. On the other hand, previous studies on motion planning at unsignalized intersection, namely in the longitudinal direction, can be categorized into three main categories: (1) Cooperative approaches; including game-theoretic, (2) Heuristic-based approaches and hybrid approaches, which combine multiple classes of these algorithms for handling the unsignalized intersection problem, and (3) Cooperative approaches entail V2V communication, which makes them unscalable based on current vehicle hardware [19], [20], [21], [22]. Game-Theoretic-Based algorithms were adopted to model the vehicles' interactions in unsignalized intersections [23], [24], assuming that the states of the interacting vehicles are observed by the subject vehicle, which allows for predicting their future trajectories and then plan its own. However, this assumption is not likely to hold for current real-life decision-making at unsignalized intersections. Heuristics-based approaches are commonly classified into two main groups: rule-based and learning-based approaches [25]. Rule-based approaches use safety intersection metrics, i.e. time-to-intersection (TTI) and time-to-collision (TTC), to constrain the commanded actions, whereas machine learning approaches, especially Reinforcement Learning (RL) approaches, focus on studying the interaction between the vehicle and the environment to learn an optimal crossing policy [26]. Unlike rule-based approaches which assume full knowledge of the environment, RL-based approaches can learn driving policies in a continuously changing environments by training their models to map the environmental observations to actions.

D. DRL for Unsignalized Intersection Traversal Problem

Deep Reinforcement Learning (DRL) techniques have been employed for deriving safe driving policies at unsignalized intersections for autonomous vehicles due to their significant capabilities in handling high-dimensional perceptual observations with discrete and continuous action spaces [27], [28]. However, the more complicated the driving scenario, the more training time is required for the DRL algorithm to converge. Hence, for learning complex behaviors efficiently with less training iterations, Curriculum Learning (CL) principles were applied while training an autonomous agent. CL was proposed in [29] as a way to accelerate learning by first training the system on simple tasks and thereafter progressively increasing the difficulty of the tasks given to the learning agent. Apart from learning high-speed autonomous overtaking [30], learning through designed curricula has also manifested significant learning benefits in terms of training time reduction and faster convergence in urban driving settings at unsignalized intersections. For instance, in [25], a curriculum DRL-based motion planning system was proposed for crossing a four-way unsignalized intersection autonomously. The proposed algorithm was designed to generate curricula in order to learn the crossing policy with fewer training iterations. However, simple one-dimensional crossing behavior is learned, while other more complex scenarios. i.e. two-dimensional left-turn was not investigated.

Designing left-turn decision-making frameworks for urban autonomous vehicles at unsignalized intersections is a highly complicated engineering problem as intersection-users turning

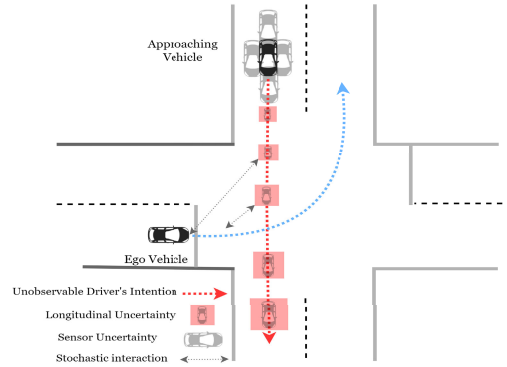


Fig. 2. An intersection-traversal scenario (left-turn) where the ego vehicle must handle several sorts of uncertainties associated with the approaching vehicle.

behaviors are not governed by traffic control signals (see Fig. 2). This problem can be modeled as a Markov Decision Process (MDP) where solutions can be obtained by utilizing online solvers or through optimal policy approximation using RL approaches. For instance, in [31] and [32], the Adaptive Belief Tree (ABT) solver is used to solve the formulated Partially-Observable Markov Decision Process (POMDP) [33] of the decision-making problem. Authors use the *Critical-Turning-Point (CTP)* approach where the left turn trajectory is simply assumed as a straight line with a quarter circle curve. However, online solvers work only for fairly small state spaces, and for larger state spaces the complexity of solving MDP scales dramatically. On the other hand, Deep Reinforcement Learning (DRL), can work with much larger, or even continuous spaces, such as Atari [34]. Furthermore, DRL approaches can also approximate their optimal policies without the requirement of observing the full state space. Considering these qualities of DRL architectures, [35], [36] introduced reinforcement-learning-based frameworks to generate safe driving policies for left-turn. However, On the behavioral planning side, the utilization of Deep Q-network (DQN) method is inefficient for urban driving environment which requires continuous actions rather than discrete ones. On the low-level side, similar to Stanley and Pure Pursuit controllers, they used geometric controller does not represent actual vehicle constraints, e.g. max steering rate, although it can minimize the tracking error. In addition, as the error does not consider error dynamics over time, their performance deteriorates significantly on high speeds which makes them suitable for low-speed maneuvers.

E. MPC for Motion Planning at Intersections

Numerous research papers have addressed the low-level motion planning problem and control at urban unsignalized intersections using Model Predictive Control principles. For instance, Hu et al. [37] proposed an event-triggered model predictive adaptive dynamic programming technique for motion planning at urban intersections. The method takes urban speed, vehicle kinematics and road constraints into consideration while solving a multi-objective optimization problem. However, for high-fidelity decision-making applications in urban autonomous driving, incorporating the local motion planning and control layers and taking into account vehicle dynamics is essential to ensure the feasibility of the high-level RL commanded actions. Such integration has been done for intersection-management applications, where centralized reference signals being distributed to the intersections agents via V2V communication [38], [39]. In [39], an integration between high-level decision-making layers and low-level MPC-based motion planing layer has been proposed for learning supervisory intersection-management policy in connected driving fashion. However, as per the authors' knowledge, such integration has not been developed for learning intersection-traversal policy of the ego vehicle agent. Hence, having the motion planning layer integrated while approximating intersection-traversal policies would facilitate learning, with feasibility guarantees [40], taking into account lateral and longitudinal dynamics.

F. Contributions

To summarize, the state-of-the-art decision-making approaches focus on advancing the high-level behavioral reasoning neglecting the importance of feasibility guarantees provided by low-level motion planning and control layer [10]. We believe that low-level motion planning integration is required to obtain efficient policies for driving in safety-critical environments [41]. Motivated by the aforementioned features of DRL architectures, particularly the soft-actor-critic (SAC) architecture [42], which has demonstrated remarkable ability in learning optimal policies for overtaking and maneuvering at roundabouts [43]. In this paper, we propose a novel hierarchical learning-based technique for left-turn decision-making at four-way unsignalized intersections. The paper offers the following main contributions:

- A novel soft-actor-critic decision-making framework is proposed, in which an integration between the behavioral planning and motion planning layers is developed to learn feasible driving policies for unsignalized intersection left-turn traversal.
- The soft-actor-critic scheme is trained using an adaptive entropy regularization technique which has resulted in faster convergence, higher mean rewards, and lower collision rates. After conducting several training experiments for the developed integrated policy, a consistent performance is observed.
- RL baselines comparison is conducted, in which we carry out several urban driving simulation experiments to evaluate the performance of the proposed integration with other model-free algorithms while accounting for real-world constraints including vehicle dynamic constraints and ride comfort.

The rest of the paper is structured as follows. Section II presents the proposed integrated decision-making scheme. Section III illustrates the experimental setup design and implementation details. Section IV presents the results of the proposed integration followed by a discussion. Finally, section V concludes the proposed and future works.

II. A Hierarchical Reinforced Learning Approach

The suggested approach has two primary planning layers: a high-level behavioral layer and a low-level motion control

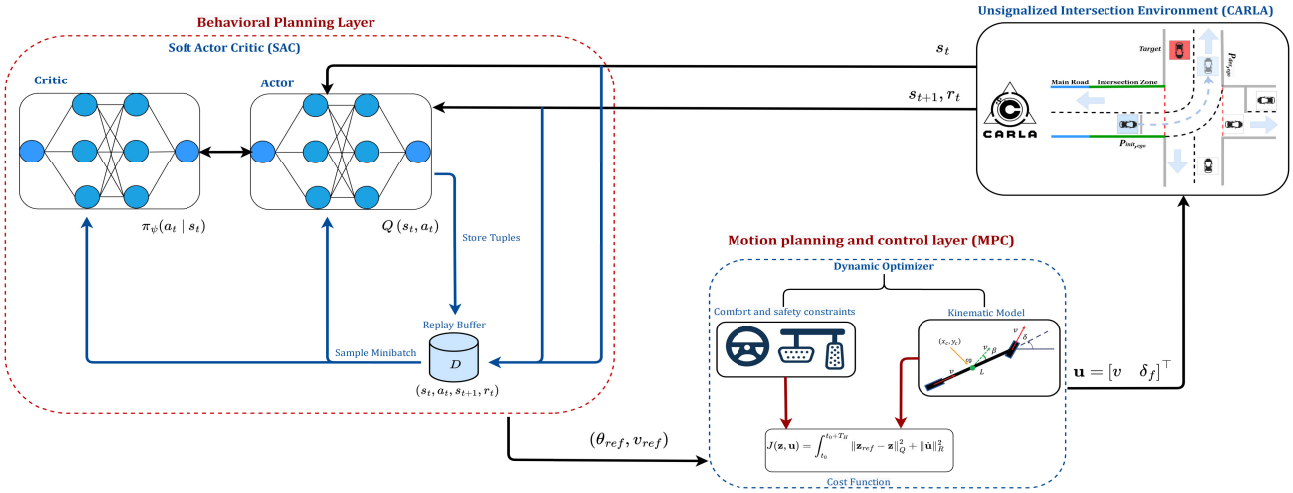


Fig. 3. An illustrative sketch of the proposed hierarchical decision-making algorithm. The agent (decision maker) is denoted by the two integrated planning layers, whereas the CARLA simulated driving scenario is the environment.

layer. The high-level behavioral planning is considered as a reinforcement learning problem, whereas the low-level motion is planned using a nonlinear model predictive control technique. The behavioural layer maps the observations collected from the driving environment into reference control signals, which are subsequently passed to MPC, which solves the tracking optimization problem and provides low-level commands to the ego vehicle. In this section, we show our multi-layered decision-making method for left-turn maneuvers at four-way unsignalized intersections. We focus on problem formulation, reward function design, and observation and action spaces. We next describe the implemented soft actor-critic principles, as well as the low-level motion planning and control, which is also presented in greater detail.

A. Overview of the Integrated Framework

We designed a targeted driving scenario, utilized in several works [25], [32], for the proposed decision-making challenge. The decision-making of the ego vehicle (in black) is coupled to the motion trajectory of the target vehicles (in red) while traversing the unsignalized intersection, as shown in Fig. 2. This scenario simulates an ego vehicle navigating in a world where the simulated target vehicles do not decelerate or yield to the ego vehicle. We further assume that the target vehicle motion is observed by the ego vehicle [25].

Fig. 3 exhibits the integrated decision-making framework. The decision-maker (agent), the ego vehicle, receives the perceptual observations and chooses actions accordingly. These actions are applied to the driving environment, and the environment returns rewards and new set of observations. As a standard RL setting, through these interactions with the driving environment, the ego-vehicle trains a policy that provides actions to maximize the future rewards. In our example, this policy is trained with the SAC algorithm to output reference velocity v_{ref} and heading signals θ_{ref}. The motion planning layer takes these reference signals as inputs to the two-dimensional tracking control problem, solving the formulated optimization problem while accounting for real-world constraints related to vehicle dynamics, urban traffic rules, and ride comfort. The optimized, feasible, control inputs are then produced to drive the vehicle's physical model in the simulated driving environment. The design process of these planning layers are explained in greater details in sections II-B and II-C.

B. High-Level Behavioral Layer

A Markov Decision Process is used to formulate the left-turn behavioral planning problem. An MDP is described as a tuple $\{S, A, R, T, \gamma\}$. The observed state S includes the ego vehicle state as well as the state of the target vehicle. Specifically, ego and target velocities, positions, and information related to lane geometry are included. The transition function T maps state-action pairs to a new state. The immediate reward is defined by the reward function R, whereas γ represents the discount factor for long-term rewards. While abiding by the speed limit, we design our learning scheme based on minimizing the left-turn time and avoiding possible collision with other intersection vehicles. Hence, the optimality of the approach can be determined by identifying the best trade-off between these two conflicting interests.

1) Reward Function: The reward function requires a significant amount of shaping to show effective learning capabilities. Given that safety is the most important factor in autonomous driving, we formed a reward function that prioritizes safety while maintaining a balance between transportation efficiency and safety during the left-turn maneuver. For the high-level two-dimensional left turn behavioral planning, the desired driving behavior of the learning agent is to proceed to the end of the route (completing the left turn) as efficiently as possible while remaining safe (in lane and no collisions). We performed extensive experimentation with several reward designs and we found that a progress-along-route reward did not lead to efficient completion because the agent was encouraged to take many steps to gather more reward. Instead, a negative reward for distance-to-goal did encourage efficient route completion. Furthermore, a positive reward is given for high speeds that remain under the speed limit of 12m/s, while negative rewards are given for exceeding the speed limit. A negative reward is given proportional to lateral deviation from the

lane center. The developed reward function is described as follows:

$$r_{ego} = r_{eff} + c_1 r_{dtg} + c_2 r_{lat} + r_{terminal} \tag{1}$$

where r_{dtg} and r_{lat} represent the distance-to-goal and lateral deviation penalties, respectively. These terms are tuned by the constants c_1 and c_2. $r_{terminal}$ is the route completion/non-completion reward/penalty, which was tuned to reward the agent if the route was successfully completed or penalise the agent if there was a collision or exceeded the maximum number of steps. The remaining term r_{eff} is designed to ensure efficient left-turn crossing where the agent is encouraged to drive as quickly as possible while remaining under the speed limit v_{lim}. The term can be described as follows:

$$r_{eff} = \begin{cases} c_3 * \left(v_{ego} - v_{lim}\right) & if \quad v_{ego} > v_{lim} \\ c_4 * v_{ego} & otherwise \end{cases} \tag{2}$$

where v_{lim} and v_{ego} represent the urban speed limit and the current ego vehicle speed, respectively. $0 \geq c_3$ is a hyperparameter for adjusting the progression penalty when $(v_{ego} \geq v_{lim})$, whereas $1 \geq c_4 \geq 0$ is the positive progression reward.

2) Observation and Action Spaces: As we assume that the target vehicle's crossing behavior is observed by the ego vehicle, the learning agent has some information about other involving agents moving within its visibility range. Hence, the observation space includes information about the dynamic states of the ego vehicle itself and the target vehicle velocity, assuming that it is traversing the intersection maximum urban speed allowed, and other input features related to the intersection geometry and collision flags. The state provided to the agent contains the ego velocity, acceleration, yaw angle delta with respect to the lane center, yaw rate, and the lateral deviation from lane center of the ego vehicle. The state also contains the yaw angle of the lane at an interval of 1, 5, and 10 meters in front of the ego, which is necessary to track the lane center. The state also contains the relative lateral and longitudinal distance between the target and ego vehicles, as well as the velocity of the target vehicle. Finally, the previous action taken is also recorded in the next state. All dynamic features in the observation vector have been normalized to ensure that they vary in identical ranges for improved convergence capabilities. Table I shows the definitions of the observation and action spaces.

3) Left-Turn Behavioral Planning Using SAC: To train the policy network, we implement SAC algorithm with adaptive exploration capability. Combining the actor-critic principle and adaptive entropy regularization, SAC trains its stochastic policy in an off-policy fashion to identify an optimal trade-off for the explore-exploit problem avoiding possible premature convergence. In addition to learning a policy π_ψ, the algorithm learns two Q-function approximators (networks) Q_{ϕ_1}, Q_{ϕ_2}, which are updated by the following loss function:

$$\mathcal{L}\left(\phi_i\right) = \underset{\left(s_t, a_t, r_t, s_{t+1}\right) \sim \mathcal{D}}{\mathbb{E}} \left[\left(Q_{\phi i}\left(s_t, a_t\right) - y\left(r_t, s_{t+1}\right) \right)^2 \right], \tag{3}$$

TABLE I
DESCRIPTION OF THE OBSERVATION AND ACTION STATES

	Observation space ($\mathbb{R}^{15}$)	
v_e	magnitude of linear velocity of the ego vehicle	$\mathbb{R}$
$\dot{v}_e$	magnitude of linear acceleration of the ego vehicle	$\mathbb{R}$
$d\theta_e$	delta yaw angle of ego vehicle	$\mathbb{R}$
$\dot{\theta}_e$	yaw rate ego vehicle	$\mathbb{R}$
d_{cl}	lateral deviation from lane center of the ego vehicle	$\mathbb{R}$
a_{t-1}	previous action	$\mathbb{R}^2$
v_{tar}	linear velocity of the target vehicle	$\mathbb{R}$
p_{tar}	longitudinal and lateral distance to the target vehicle	$\mathbb{R}^2$
ψ_l	yaw angle of the lane at 1, 5, 10 meters ahead	$\mathbb{R}^3$
	Action space ($\mathbb{R}^2$)	
v_{ref}	reference velocity signal	$\mathbb{R}$
θ_{ref}	reference heading signal	$\mathbb{R}$

where $i = 1, 2$ and $\mathcal{D}$ represents the experiences obtained by the agent's while exploring the environment. The target $y\left(r_t, s_{t+1}\right)$ is given by:

$$y\left(r_t, s_{t+1}\right) = r(s_t, a_t) + \gamma T_{tar}(s_{t+1}), \tag{4}$$

where T_{tar} function is optimized by the following equation:

$$T_{tar}(s_{t+1}) = \left(\min_{j=1,2} Q_{\phi_{\text{tar},j}}\left(s_{t+1}, \tilde{a}_{t+1}\right) - \alpha \log \pi_\psi\left(\tilde{a}_{t+1} \mid s_{t+1}\right) \right), \tag{5}$$

where $\tilde{a}_{t+1}$ represent the next actions which are sampled from the policy $\pi_\psi\left(\cdot \mid s_{t+1}\right)$. α is used the tune exploitation-exploration trade-off which is governed by the policy's entropy. For instance, decreasing α values results in less exploration. In our implementation we schedule the entropy regularization coefficient α based on the performance of the agent. Specifically, we use the following equation to adjust the exploration-exploitation term based on the performance of the agent [30]:

$$\alpha = clip(1 - \lambda, v_1, v_2), \tag{6}$$

where λ is the success rate over the past 10 evaluation episodes, and the hyperparameters v_1 and v_2 are tuned to be 0.1 and 0.3, respectively. We start with $\alpha = 0.3$ and to ensure that the agent sufficiently explores the environment avoiding any local optimum. As shown in Fig. 5, adaptive entropy regularization has resulted in faster convergence, higher mean rewards and less collision rates.

The policy is learnt by maximizing the expected future return and expected future entropy as denoted in $V^\pi(s_t)$ function:

$$\begin{aligned} V^\pi(s_t) &= \underset{a_t \sim \pi}{\mathrm{E}} \left[Q^\pi(s_t, a_t) \right] + \alpha H(\pi(\cdot \mid s_t)) \\ &= \underset{a_t \sim \pi}{\mathrm{E}} \left[Q^\pi(s_t, a_t) - \alpha \log \pi(a_t \mid s_t) \right]. \end{aligned} \tag{7}$$

Following the current policy, the actions can be derived from the current policy $\tilde{a}' \sim \pi_\psi\left(\cdot \mid s\right)$, whereas the states are drawn from the replay buffer $s \sim \mathcal{D}$. We use the reparameterization trick to optimise the policy $\pi_\psi\left(\cdot \mid s\right)$, in which a sample from actions is drawn by computing a deterministic function of

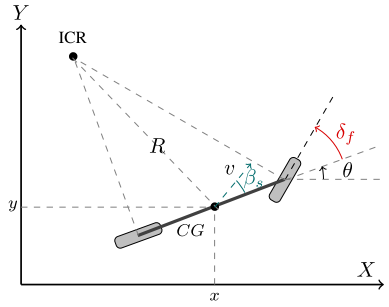


Fig. 4. Kinematic bicycle model schematic.

state, policy parameters, and independent noise ξ as follows:

$$\tilde{a}_\psi(s) = \tanh\left(\mu_\psi(s) + \sigma_\psi(s) \odot \xi\right), \quad \xi \sim \mathcal{N}(0, I) \tag{8}$$

where tanh is used to bound the obtained actions to a finite range of $[-1, 1]$.

The loss function for SAC learning, can be formulated as to maximize the expected future rewards plus the expected future entropy as can be described below:

$$\mathcal{L}(\psi)_{SAC} = \mathbb{E}_{s_t \sim \mathcal{D}}\left[\alpha \log \pi_\psi\left(\tilde{a}_\psi(s_t) \mid s_t\right) - \min_{i=1,2} Q_{\phi_i}\left(s_t, \tilde{a}_\psi(s_t)\right)\right] \tag{9}$$

C. Low-Level Motion Planning and Control Layer

The low-level layer is based on Model Predictive Control (MPC), that is responsible for vehicle motion control. The MPC controller respects the vehicle non-holonomic constraints by utilizing vehicle kinematic model for prediction. The model is shown in Fig. 4 and stated below.

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = f\left(\mathbf{x}(t), \mathbf{u}(t)\right) = \begin{bmatrix} v\cos(\theta + \beta_s) \\ v\sin(\theta + \beta_s) \\ \dfrac{v\cos(\beta_s)\tan(\delta_f)}{L} \end{bmatrix}, \tag{10}$$

where

$$\beta_s = \arctan\left(\frac{l_r \tan\delta_f}{L}\right) \tag{11}$$

where the vehicle state is $\mathbf{x} = \begin{bmatrix} x & y & \theta \end{bmatrix}^\top$, x and y are the position of the vehicle in X-Y global frame, and θ is the vehicle orientation in the global frame. Furthermore, $\mathbf{u} = \begin{bmatrix} v & \delta_f \end{bmatrix}^\top$ is the vector of control actions, v is the velocity of the ego vehicle at its C.G., and δ_f is the steering angle. In Fig. 4, β_s is the side-slip angle of the vehicle, l_r is the distance between the rear axle and the C.G., and L is the wheelbase length of the vehicle.

The intuition behind using a vehicle kinematic model for optimization is that the dynamic effects are negligible due to low speed driving in urban environments. In addition, to ensure ride comfort and safety, several hard constraints are set on the optimization variables as given in table II.

To let the high-level control layer drive the ego vehicle while abiding vehicle constraints and respecting the static map, e.g. road network, the MPC objective function is formulated to minimize the following running costs:

TABLE II
CONSTRAINTS SET ON THE OPTIMIZATION VARIABLES

Parameter	Lower Bound	Upper Bound
$\dot{\mathbf{u}}(t)$	$\dot{\mathbf{u}}(t)_{min} = \begin{bmatrix} -3\ \text{m/s}^2 \\ -\frac{\pi}{3}\ \text{rad/s} \end{bmatrix}$	$\dot{\mathbf{u}}(t)_{max} = \begin{bmatrix} 5\ \text{m/s}^2 \\ \frac{\pi}{3}\ \text{rad/s} \end{bmatrix}$
$\mathbf{u}(t)$	$\mathbf{u}(t)_{min} = \begin{bmatrix} -2.25\ \text{m/s} \\ -\frac{\pi}{3}\ \text{rad} \end{bmatrix}$	$\mathbf{u}(t)_{max} = \begin{bmatrix} 12\ \text{m/s} \\ \frac{\pi}{3}\ \text{rad} \end{bmatrix}$

1) Velocity error between reference and actual speed of the vehicle.
2) Heading error between reference and actual heading of the vehicle.
3) Change in control actions.

Velocity and heading errors are based on the reference generated by the high-level control layer. Therefore, the high-level agent learns to generate the appropriate heading reference and velocity reference, and the low-level MPC achieves them while satisfying vehicle constraints. Therefore, the objective function J is given by:

$$J\left(\mathbf{z}, \mathbf{u}\right) = \int_{t_0}^{t_0+T_H} \left|\left|\mathbf{z}_{ref} - \mathbf{z}\right|\right|_Q^2 + \left|\left|\dot{\mathbf{u}}\right|\right|_R^2 \tag{12}$$

where $\mathbf{z} = \begin{bmatrix} \theta & v \end{bmatrix}^\top$ and, $Q \in \mathbb{R}^{2\times2}$, $R \in \mathbb{R}^{2\times2}$, t_0 is the initial time, and T_H is the prediction horizon. Accordingly, the Optimal Control Problem (OCP) can be formulated as follows:

$$\min_{\mathbf{u}(t)} \ J\left(\mathbf{z}(t), \mathbf{u}(t)\right) \tag{13a}$$

$$\text{s.t. } \dot{\mathbf{x}}(t) = f\left(\mathbf{x}(t), \mathbf{u}(t)\right), \quad \forall t \in [t_0, t_0 + T_H] \tag{13b}$$

$$\dot{\mathbf{u}}_{min}(t) \le \dot{\mathbf{u}}(t) \le \dot{\mathbf{u}}_{max}(t), \quad \forall t \in [t_0, t_0 + T_H] \tag{13c}$$

$$\mathbf{u}_{min}(t) \le \mathbf{u}(t) \le \mathbf{u}_{max}(t), \quad \forall t \in [t_0, t_0 + T_H] \tag{13d}$$

Every timestep, the high-level behavioral layer generates a new reference velocity and a new reference heading. The values are absolute and not the difference from current state of the vehicle. The low-level layer then takes these values as references and solves the OCP for the optimal control actions.

For more details, the implementation of the proposed integration between the behavioral and motion planning layers is described with greater details in algorithm 1.

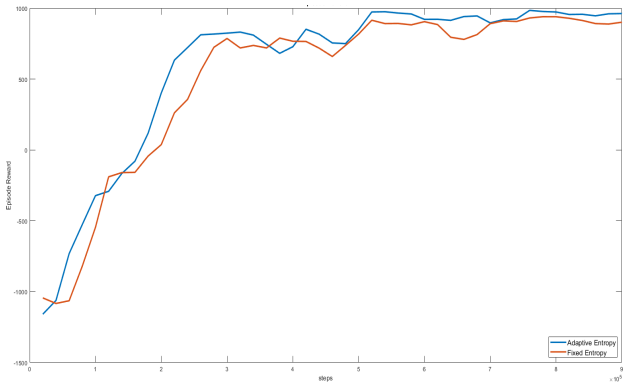
III. EXPERIMENTS

A. Environment Setup and Implementation Details

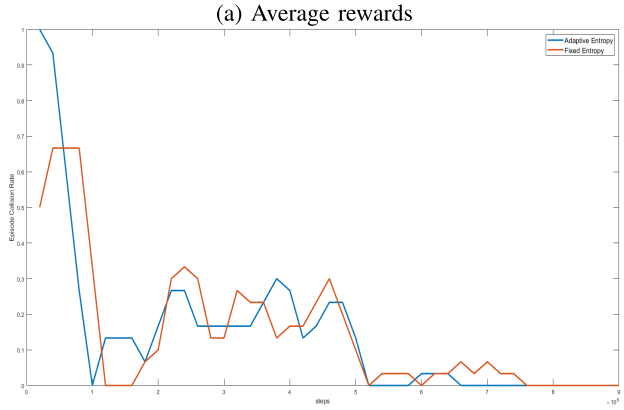
We designed our reinforcement learning environment using the CARLA simulator [44] as it provides realistic simulated driving scenarios within urban environments, in addition to its flexible Python API. Fig. 6(a), shows the task setup. We spawn the autonomous vehicle at a fixed predefined starting point aiming at learning how to follow the planned left-turn route efficiently and safely (no collision with target vehicle, no lane invasions), and reaching the predefined destination point. The target vehicle is spawned randomly covering all possible locations along the route as shown in Fig. 6(b). We initiate up to 40 CARLA instances, simulating the same left-turn driving environment, which results in faster experience sampling compared to a single CARLA instance.

Algorithm 1 SAC With MPC Integration

Initialize policy parameters θ, Q-function networks (Q_{ϕ_1}, Q_{ϕ_2}), empty replay buffer $\mathcal{D}$
for step in warm-up steps **do**
    Run network with randomized weights
end for
repeat
    Observe state s_t and sample action $\tilde{a}_\psi$
    Pass action $\tilde{a}_\psi$ to MPC as $\mathbf{z}_{ref}$
    MPC OCP is solved for optimal control actions while respecting vehicle dynamics and ride comfort using Eq.13
    Execute action **u** in the environment
    Observe next state s_{t+1} and obtain reward r_t using Eq.1 and Eq.2
    Increment total steps taken
    if Memory buffer $\mathcal{D}$ is full **then**
        Delete oldest transition in buffer
    end if
    Store transition (s_t, a_t, s_{t+1}, r_t) in memory buffer $\mathcal{D}$
    if time to update **then**
        Randomly sample mini-batch from replay buffer $\mathcal{D}$
        Update Q network policy parameters Q_{ϕ_1}, Q_{ϕ_2} using Eq.3 and Eq.4
        Update entropy bonus parameter α using Eq.6
    end if
    if s_{t+1} is terminal state **then**
        Reset environment
    end if
until Convergence or current step is equal to max steps

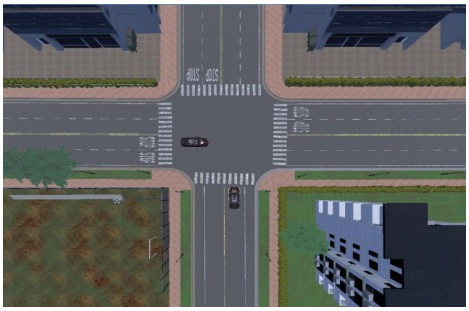


(a) Average rewards

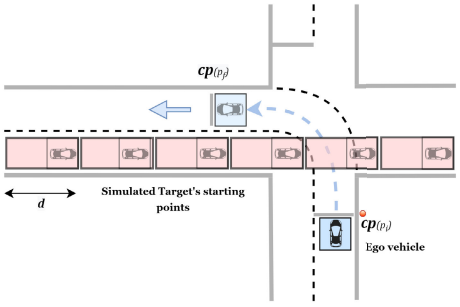


(b) Collision rates

Fig. 5. Comparison of training curves of driving polices (with and without adaptive entropy regularization).



(a)



(b)

Fig. 6. Driving experiment setup. Fig 6(a) illustrates the CARLA unsignalized intersection environment setup. Fig. 6(b) shows the same setup with the possible starting points of the target vehicle.

We make use of Baidus deep learning framework-PaddlePaddle which is available for both CPUs and GPUs. This forms the base of a high-efficient reinforcement learning framework, PARL [45]. This framework provides several research capabilities including reusability, reproducibility and extensibility. Building and integration of custom algorithms for policy training becomes feasible via use of this framework. It is composed of three major components - Agent, Algorithm and Model. The algorithm represents the update mechanism for parameters in model and therefore necessarily contains one model which abstracts the forward network. The model via abstraction delineates the forwards network defining critic or policy network which accepts states as input. The agent forms the data bridge for data I/O between the environment and algorithms. These three components serve as a compact API for distributed training by addition of a decorator.

Our customized decision-making reinforcement learning agent is implemented using PARL. We have chosen PARL due to its capability in supporting high-performance training parallelization with large number of CPUs and multi-GPUs, which is necessary to collect a large volume of experiences on the relatively slow Carla simulator. Additionally, PARL offers existing implementations of popular model free algorithms (TD3, PPO, SAC) on the Mujoco task set, which we were able to adapt to the CARLA-based environment we propose. The hyper-parameters used for the parallel version of SAC have been listed in Table III.The simulation timestep is 0.1 second, and the high-level and low-level layers run at 10 Hz and 20 Hz, respectively.

TABLE III
EXPERIMENT PARAMETERS

Hyperparameter	Value
NN size	6X [256, ReLu]
Mini batch size	512
Replay buffer size	5e+05
Actor Learning Rate	3e-04
Critic Learning Rate	3e-04
Exponential Discount Factor	0.99
Max Episode Steps	500

B. Policy Training and Evaluation

The SAC neural network is randomly initialized and trained via maximizing the left-turn reward function explained in (Eq. 1). We defined the stopping criteria for any training episode to be in cases where the agent exceeds the predefined maximum number of steps, reaches the destination, collides with the target vehicle, or lateral deviates from lane center more than 7.5 meters. The target vehicle is programmed to not yield to the ego vehicle as well as to abide by the urban speed limit.

To provide a holistic evaluation of the proposed design performance, we compare it to other model-free DRL techniques that can handle the continuous action space of the problem. In this study, we test the Twin Delayed Deep Deterministic Policy Gradient (TD3) and proximal policy optimization (PPO) approaches, which are common baselines for RL policy learning comparison in autonomous vehicles [30]. TD3 [46] is a modified, state of the art actor-critic, Deep Deterministic Policy Gradient algorithm for problems with continuous control domains. On the other hand, PPO is an improved on-policy of Trust Region Policy Optimization (TRPO) for robotics and games playing applications [47].

IV. RESULTS AND DISCUSSION

In this section, we present the results of the proposed decision-making approach for left-turn traversing behaviour at four-way unsignalized Intersection. in sectionIV-A, we demonstrate the high-level behavioral planning layer performance using several model-free DRL algorithms, whereas the results of the integrated scheme are highlighted in section IV-C. We also conclude this section with remarks highlighted in section IV-D.

A. Model-Free Behavioral Planning Comparison

We compare the learning performance of SAC with other model-free RL algorithms, namely TD3 and PPO, in this section. For this comparison, the low-level motion planning layer is not integrated, the decision-maker (agent) receives observations from the intersection driving environment and maps them directly into throttle and steering commands executed by the environment. As seen in Fig. 7(a), SAC outperforms both TD3 and PPO in terms of maximizing cumulative reward with fewer samples. SAC and TD3 do achieve a higher reward compared to PPO, this can be attributed to the entropy bonus in SAC allowing it to discover a better policy via exploration and double-critic architecture in SAC which improves

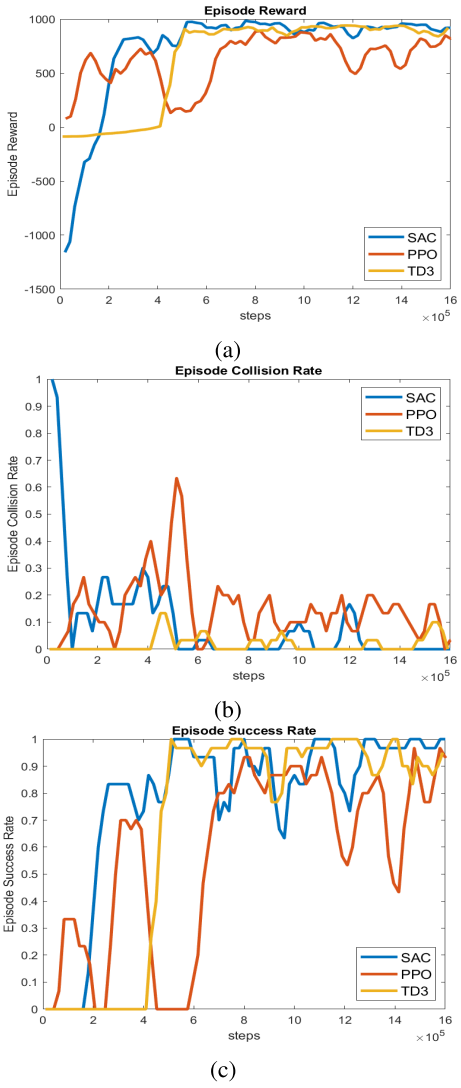


Fig. 7. Model-free comparisons. Fig. 7(a) and Fig. 7(b) show the average episodic reward and collision rate, respectively. Fig. 7(c) represents the average success rate. The models are trained until their performance converged.

the learning performance by reducing the overestimation bias. Furthermore, PPO takes the largest number of policy updates to converge but less consistently, as shown in Fig. 7(b), where collisions occur with higher rate compared to SAC and TD3.

B. Learning Performance in More Complicated Scenes

A set of several experiments have been conducted to demonstrate the learning capability in more complicated scenes with more target vehicles. We expanded the simulation environment by including a second target-vehicle going right and sharing the lane with the ego vehicle (see Fig.8).

We modified the state space by adding the motion states of the second target vehicle as it moves within the ego vehicle's range of visibility, and the reward function was adjusted to include an additional crash penalty. To compare with the one-target vehicle simulation results, we ran this experiment for the same number of training steps.

As shown in Fig.9(a), complicating the scene by adding more target-vehicles, as expected, necessitates more steps for the policy to learn a traversing behaviour. The results show that

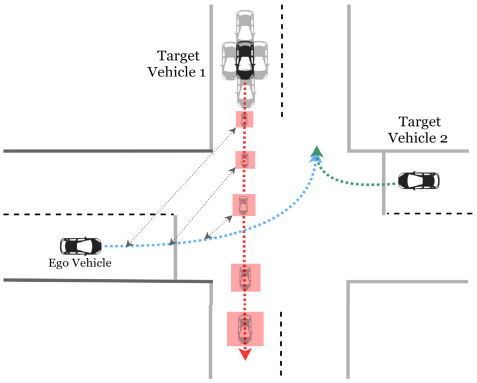


Fig. 8. An Illustration of intersection-traversal scenario (left-turn) where the ego vehicle negotiates with two target vehicles.

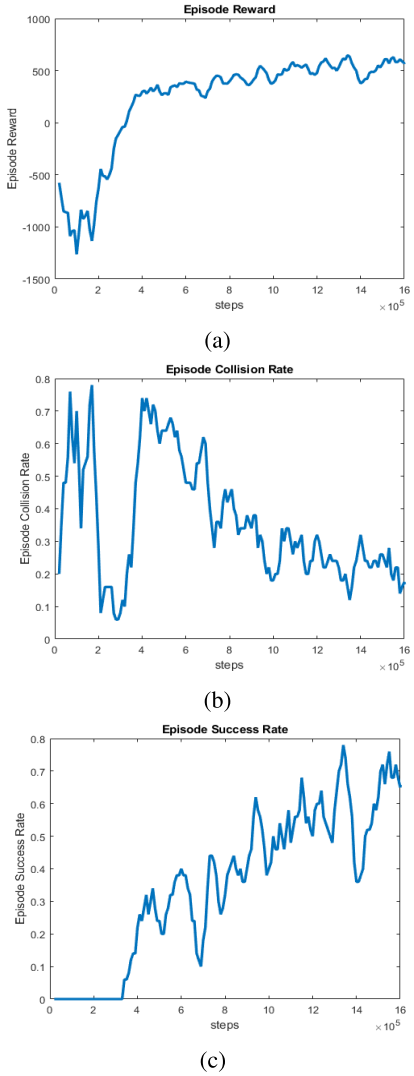


Fig. 9. Traversing with two-target vehicles experiment. Fig. 9(a) and Fig. 9(b) show the average episodic reward and collision rate, respectively. Fig. 9(c) represent the average success rate. The models are trained until their performance converged.

the policy has converged to lower average rewards when compared to the single-target vehicle settings, owing to the longer time required to complete the turn. Figures 9(b) and 9(c) show that as training progresses, both collision and success rate

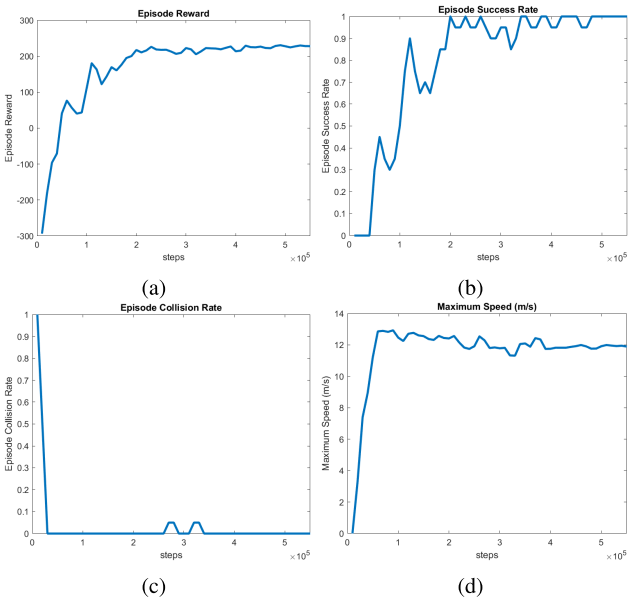


Fig. 10. Training curves of the proposed reinforced learning hierarchical decision-making scheme. Fig. 10(a) illustrates the average episodic reward. Fig. 10(b) and 10(c) exhibit the success rate and the collision rate, respectively. The maximum episodic speed is plotted in Fig. 10(d) where the agent converges to the max urban speed allowed (12 m/s). The Policy is trained until convergence.

improve, demonstrating the learning capability and design flexibility of the developed behavioural planner.

C. Integrated Scheme Results

In this section, we illustrate the results of the proposed hierarchical decision-making scheme at unsignalized intersections. These results demonstrate the effectiveness of learning efficient, yet safe, left-turn behaviors with feasibility guarantees. The training performance of the network policy is evaluated using predefined Key Performance Indicators (KPIs). Among these, the average episodic reward, the average success rate, the average collision rate and the max episodic speed. Fig. 10 exhibits the training evaluation for the first 500k training steps.

As shown in Fig. 10, at the very beginning of the training process, the average reward is noticeably low, which means that the agent is being heavily penalized as the collision rate is very high which hinders the agent from completing the task. As the training passes 30k steps, the agent significantly learns how to avoid colliding with the target vehicle as Fig. 10 shows a significant quick drop in the collision rate which align with the significant ascension of the average reward. However, the success rate still ranges between 30% to 40% agent which shows that the policy is not trained sufficiently. The policy starts its convergence after approximately 200k steps where the average reward converges to a high value and the average success rate fluctuates with values above 90%. Fig. 10(d) illustrates the max episodic speed values during the training process. It can be discerned that the proposed decision-making approach provides feasible actions that abide by the constraints and ride comfort with a success rate above 90%.

The same training scenario, where the ego vehicle embarks on its two dimensional motion at the stop line and the target

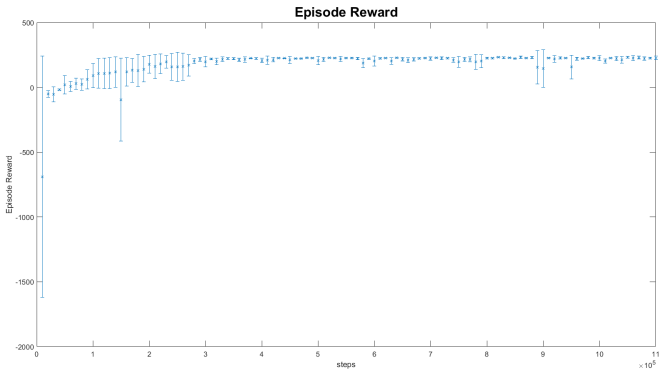
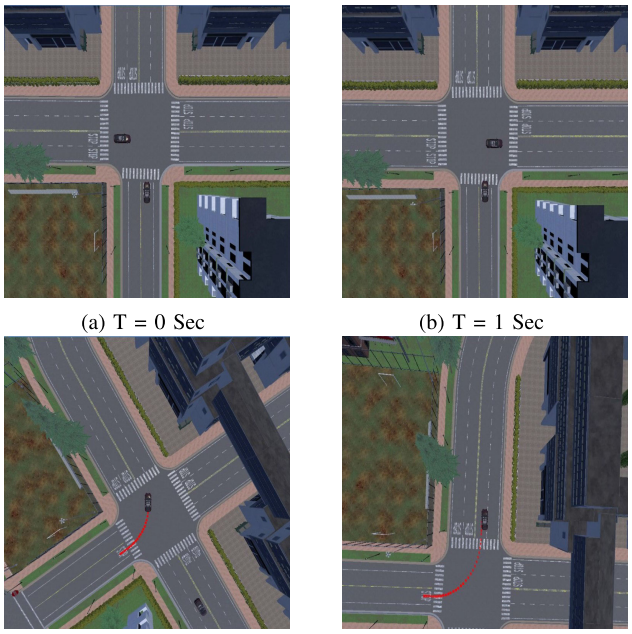


Fig. 11. Mean of the average rewards (denoted by x) and standard deviations of several training experiments of the developed integrated model.



(a) T = 0 Sec (b) T = 1 Sec

(c) T = 4 Sec (d) T = 6 Sec

Fig. 12. Testing snippets of left-turn maneuvers at unsignalized Intersection.

vehicle is spawned randomly in the scene, is adopted for testing the trained decision-making model. After conducting several training experiments for the developed integrated policy, a consistent performance is observed (see Fig. 11). We save the superior trained policy to test the performance of the left-turn behavior over 1000 episodes. The results show that the agent can maneuver left-turns with a success rate of 97.8%, colliding only once with the target vehicle, and failing to complete the task successfully in 21 coincidences (due to exceeding the maximum number of steps or lane departure). The effectiveness of the learnt traversal policy is visually demonstrated in Fig. 12, where we show a left-turn maneuver for one of the challenging traversal scenarios where the target vehicle shares conflict points with the ego vehicle. As seen in Fig. 12(a), the ego is at the stop line waiting for the target vehicle to traverse the intersection environment. The non-cautious, yet safe, behavior of the trained policy can be shown in Fig. 12(b), which is snipped after 1 sec, where the ego vehicle starts it is two-dimensional motion as quickly as safely possible. Fig. 12(c) and Fig. 12(d) represent the left-turn progress when nearly and fully completed, respectively.

In terms of computation complexity, the proposed integrated scheme makes use of two CPU cores (on an AMD Ryzen Threadripper PRO 3975WX) and 250MB of RAM. Practically, in real AVs, such as the Chevrolet Bolt used in the SAE AutoDrive Challenge, we had 64 CPU cores and 128GB of RAM, so on that vehicle deployment, the integration could use 3% of the CPU and 0.2% of the available RAM, which is considered marginal.

D. Discussion

It can be noticed from the results provided in section IV-C that learning driving policies with low-level layer integrated, is necessary for learning high-fidelity feasible driving behaviors. With the MPC enabled, the policy start converging after converges after 200k training steps with success rate 1 as depicted in Fig. 10(a) and Fig. 10(b), whereas the standalone SAC converges after 500k steps. This could be attributed to the fact that the control inputs produced by the MPC, to the environment, have been already optimized taking real-life driving constraints into account. Comparing our results with the random curriculum learning-based results provided in [25], we can discern that our work shows superiority in terms of the KPIs discussed in previous sections. Despite the fact that we tackle a challenging two-dimensional left-turn intersection-traversal, the integration between SAC-based behavioral path planning layer and MPC-based motion planning layer results in faster convergence and higher success rate compared to their one-dimensional intersection-traversing model. However, we should acknowledge that there are few limitations of the proposed work. Among these, first, the ability of handling noisy perceptual observations. Second, we need to improve the model's accuracy and navigation capabilities under occlusions where the intersection environment is partially observable. Therefore, we have future research directions towards improving the model in these aspects.

V. Conclusion

In this paper, a novel hierarchical reinforcement learning-based decision-making scheme is proposed for automated left-turn maneuvers at unsignalized intersections. The proposed novel integrated scheme combines soft-actor-critic and model predictive control principles for high-level behavioral planning and low-level motion planning layers, respectively. The goal of this integration is to learn high-fidelity left-turn behaviours while accounting for real-world constraints related to vehicle dynamics, urban traffic rules, and ride comfort. For adaptive exploration-exploitation capabilities, we modify the SAC implementation by linking the entropy bonus updates to the agent's episodic success rate. A customized CARLA urban driving environment is designed to validate the proposed decision-making scheme. The high-level training comparison shows a superiority of SAC over other model-free learning schemes including TD3 and PPO. Moreover, the training results of the integrated framework illustrates the effectiveness of the proposed method in terms of performance and sample efficiency. Finally, the testing visual demonstration illustrates

the efficiency and safety of the learn left-turn behaviors yielding a success rate of 97.8% over 1000 testing episodes.

REFERENCES

[1] Y. Kuwata, S. Karaman, J. Teo, E. Frazzoli, J. P. How, and G. Fiore, "Real-time motion planning with applications to autonomous urban driving," *IEEE Trans. Control Syst. Technol.*, vol. 17, no. 5, pp. 1105–1118, Sep. 2009.

[2] X. Li, Z. Sun, D. Cao, Z. He, and Q. Zhu, "Real-time trajectory planning for autonomous urban driving: Framework, algorithms, and verifications," *IEEE/ASME Trans. Mechatronics*, vol. 21, no. 2, pp. 740–753, Apr. 2016.

[3] P. Hang, C. Huang, Z. Hu, and C. Lv, "Driving conflict resolution of autonomous vehicles at unsignalized intersections: A differential game approach," *IEEE/ASME Trans. Mechatronics*, vol. 27, no. 6, pp. 5136–5146, Dec. 2022.

[4] C. Hu, S. Hudson, M. Ethier, M. Al-Sharman, D. Rayside, and W. Melek, "Sim-to-real domain adaptation for lane detection and classification in autonomous driving," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2022, pp. 457–463.

[5] J. Zhao, V. L. Knoop, and M. Wang, "Microscopic traffic modeling inside intersections: Interactions between drivers," *Transp. Sci.*, vol. 57, no. 1, pp. 135–155, Jan. 2023.

[6] J. Zhao, X. Yang, and C. Zhang, "Vehicle trajectory reconstruction for intersections: An integrated wavelet transform and Savitzky-Golay filter approach," *Transportmetrica A, Transp. Sci.*, pp. 1–24, Jan. 2023, doi: 10.1080/23249935.2022.2163207.

[7] R. Dempster, M. Al-Sharman, Y. Jain, J. Li, D. Rayside, and W. Melek, "DRG: A dynamic relation graph for unified prior-online environment modeling in urban autonomous driving," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2022, pp. 8054–8060.

[8] J. Liu, Y. Luo, H. Xiong, T. Wang, H. Huang, and Z. Zhong, "An integrated approach to probabilistic vehicle trajectory prediction via driver characteristic and intention estimation," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Oct. 2019, pp. 3526–3532.

[9] Z. Li, J. Gong, C. Lu, and Y. Yi, "Interactive behavior prediction for heterogeneous traffic participants in the urban road: A graph-neural-network-based multitask learning framework," *IEEE/ASME Trans. Mechatronics*, vol. 26, no. 3, pp. 1339–1349, Jun. 2021.

[10] B. Paden, M. Cáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Trans. Intell. Vehicles*, vol. 1, no. 1, pp. 33–55, Mar. 2016.

[11] R. Dempster, M. Al-Sharman, D. Rayside, and W. Melek, "Real-time unified trajectory planning and optimal control for urban autonomous driving under static and dynamic obstacle constraints," 2022, *arXiv:2209.09320*.

[12] D. Kim, B. Kim, T. Chung, and K. Yi, "Lane-level localization using an AVM camera for an automated driving vehicle in urban environments," *IEEE/ASME Trans. Mechatronics*, vol. 22, no. 1, pp. 280–290, Feb. 2017.

[13] M. Al-Sharman et al., "A sensorless state estimation for a safety-oriented cyber-physical system in urban driving: Deep learning approach," *IEEE/CAA J. Autom. Sinica*, vol. 8, no. 1, pp. 169–178, Jan. 2021.

[14] E. Namazi, J. Li, and C. Lu, "Intelligent intersection management systems considering autonomous vehicles: A systematic literature review," *IEEE Access*, vol. 7, pp. 91946–91965, 2019.

[15] M. Zhou, Y. Yu, and X. Qu, "Development of an efficient driving strategy for connected and automated vehicles at signalized intersections: A reinforcement learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 1, pp. 433–443, Jan. 2020.

[16] Y. Liu, P. Zhao, D. Qin, G. Li, Z. Chen, and Y. Zhang, "Driving intention identification based on long short-term memory and a case study in shifting strategy optimization," *IEEE Access*, vol. 7, pp. 128593–128605, 2019.

[17] G. S. Aoude, B. D. Luders, K. K. H. Lee, D. S. Levine, and J. P. How, "Threat assessment design for driver assistance system at intersections," in *Proc. 13th Int. IEEE Conf. Intell. Transp. Syst.*, Sep. 2010, pp. 1855–1862.

[18] Y. Jeong, S. Kim, and K. Yi, "Surround vehicle motion prediction using LSTM-RNN for motion planning of autonomous vehicles at multi-lane turn intersections," *IEEE Open J. Intell. Transp. Syst.*, vol. 1, pp. 2–14, 2020.

[19] M. R. Hafner, D. Cunningham, L. Caminiti, and D. Del Vecchio, "Cooperative collision avoidance at intersections: Algorithms and experiments," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1162–1175, Sep. 2013.

[20] T. Wu, M. Jiang, and L. Zhang, "Cooperative multiagent deep deterministic policy gradient (CoMADDPG) for intelligent connected transportation with unsignalized intersection," *Math. Problems Eng.*, vol. 2020, pp. 1–12, Jul. 2020.

[21] Y. Chen, J. Zha, and J. Wang, "An autonomous T-intersection driving strategy considering oncoming vehicles based on connected vehicle technology," *IEEE/ASME Trans. Mechatronics*, vol. 24, no. 6, pp. 2779–2790, Dec. 2019.

[22] P. Hang, C. Huang, Z. Hu, and C. Lv, "Decision making for connected automated vehicles at urban intersections considering social and individual benefits," 2022, *arXiv:2201.01428*.

[23] R. Tian, N. Li, I. Kolmanovsky, Y. Yildiz, and A. R. Girard, "Game-theoretic modeling of traffic in unsignalized intersection network for autonomous vehicle control verification and validation," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 2211–2226, Mar. 2022.

[24] N. Li, Y. Yao, I. Kolmanovsky, E. Atkins, and A. R. Girard, "Game-theoretic modeling of multi-vehicle interactions at uncontrolled intersections," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 2, pp. 1428–1442, Feb. 2022.

[25] Z. Qiao, K. Muelling, J. M. Dolan, P. Palanisamy, and P. Mudalige, "Automatically generated curriculum based reinforcement learning for autonomous vehicles in urban environment," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 1233–1238.

[26] D. Isele, R. Rahimi, A. Cosgun, K. Subramanian, and K. Fujimura, "Navigating occluded intersections with autonomous vehicles using deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 2034–2039.

[27] T. de Bruin, J. Kober, K. Tuyls, and R. Babuška, "Integrating state representation learning into deep reinforcement learning," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 1394–1401, Jul. 2018.

[28] J. Chen, T. Shu, T. Li, and C. W. de Silva, "Deep reinforced learning tree for spatiotemporal monitoring with mobile robotic wireless sensor networks," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 50, no. 11, pp. 4197–4211, Nov. 2020.

[29] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 41–48.

[30] Y. Song, H. Lin, E. Kaufmann, P. Duerr, and D. Scaramuzza, "Autonomous overtaking in gran turismo sport using curriculum reinforcement learning," 2021, *arXiv:2103.14666*.

[31] K. Shu et al., "Autonomous driving at intersections: A critical-turning-point approach for left turns," 2020, *arXiv:2003.02409*.

[32] K. Shu et al., "Autonomous driving at intersections: A behavior-oriented critical-turning-point approach for decision making," *IEEE/ASME Trans. Mechatronics*, vol. 27, no. 1, pp. 234–244, Feb. 2022.

[33] H. Kurniawati and V. Yadav, "An online POMDP solver for uncertainty planning in dynamic environment," in *Robotics Research*. Cham, Switzerland: Springer, 2016, pp. 611–629.

[34] M. Igl, L. Zintgraf, T. A. Le, F. Wood, and S. Whiteson, "Deep variational reinforcement learning for POMDPS," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 2117–2126.

[35] F. Wang, D. Shi, T. Liu, and X. Tang, "Decision-making at unsignalized intersection for autonomous vehicles: Left-turn maneuver with deep reinforcement learning," 2020, *arXiv:2008.06595*.

[36] H. Shu, T. Liu, X. Mu, and D. Cao, "Driving tasks transfer using deep reinforcement learning for decision-making of autonomous vehicles in unsignalized intersection," *IEEE Trans. Veh. Technol.*, vol. 71, no. 1, pp. 41–52, Jan. 2022.

[37] C. Hu, L. Zhao, and G. Qu, "Event-triggered model predictive adaptive dynamic programming for road intersection path planning of unmanned ground vehicle," *IEEE Trans. Veh. Technol.*, vol. 70, no. 11, pp. 11228–11243, Nov. 2021.

[38] K. Wang, Y. Wang, L. Wang, H. Du, and K. Nam, "Distributed intersection conflict resolution for multiple vehicles considering longitudinal-lateral dynamics," *IEEE Trans. Veh. Technol.*, vol. 70, no. 5, pp. 4166–4177, May 2021.

[39] A. H. Hamouda, D. M. Mahfouz, C. M. Elias, and O. M. Shehata, "Multi-layer control architecture for unsignalized intersection management via nonlinear MPC and deep reinforcement learning," in *Proc. IEEE Int. Intell. Transp. Syst. Conf. (ITSC)*, Sep. 2021, pp. 1990–1996.

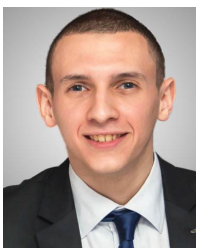
[40] M. A. Daoud, M. W. Mehrez, D. Rayside, and W. W. Melek, "Simultaneous feasible local planning and path-following control for autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 16358–16370, Sep. 2022.

[41] J. Hawke et al., "Urban driving with conditional imitation learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 251–257.

[42] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1861–1870.
[43] H. Liu, Z. Huang, J. Wu, and C. Lv, "Improved deep reinforcement learning with expert demonstrations for urban autonomous driving," 2021, *arXiv:2102.09243*.
[44] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proc. 1st Annu. Conf. Robot Learn.*, vol. 78, Nov. 2017, pp. 1–16.
[45] PARL Developers. (2021). *Parl*. [Online]. Available: https://github.com/PaddlePaddle/PARL
[46] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1587–1596.
[47] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

Mohammad Al-Sharman (Graduate Student Member, IEEE) received the B.Sc. degree in mechatronics engineering from Tafila Technical University in 2011 and the M.Sc. degree in mechatronics engineering from the American University of Sharjah in 2015. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the University of Waterloo, Canada. From 2015 to 2017, he was involved in several research activities where auto takeoff and precision landing using integrated sensors were developed for rotary wing UAVs. From 2017 to 2018, he was a Research Associate with the Robotics Institute, Khalifa University, United Arab Emirates. His current research interests include machine learning, automated driving, decision-making, reinforcement learning, and stochastic estimation. He has been a recipient of multiple academic scholarships and awards during his undergraduate and graduate studies. He is currently a Reviewer of IEEE/ASME TRANSACTIONS ON MECHATRONICS, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, and IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS.

Rowan Dempster received the B.C.S. degree from the University of Waterloo in 2020, where he is currently pursuing the M.A.Sc. degree in electrical and computer engineering (ECE) with WATonomous. He has worked on automated driving systems at WATonomous since 2017. He has recently published his environment modeling work in ICRA 2022. His current research interests include graphical environment models for decision-making, action classification in video streams, and predictive control schemes. He was a recipient of the NSERC Alexander Graham Bell Canada Graduate Scholarship.



Mohamed A. Daoud received the B.Sc. degree in mechatronics engineering and automation from Ain Shams University, Egypt, in 2018, and the M.A.Sc. degree in mechanical and mechatronics engineering from the University of Waterloo, ON, Canada, in 2020. He is currently a Software Engineer with General Motors, ON, Canada. His current research interests include self-driving cars, artificial intelligence, sensor fusion, and control theory.

Mahmoud Nasr received the B.Sc. degree in aerospace engineering from the Zewail City of Science and Technology, Egypt, in 2018, and the M.A.Sc. degree in pattern analysis and machine intelligence from the Department of Electrical and Computer Engineering, University of Waterloo, ON, Canada, in 2020. He is currently an Autonomous Robotics Engineer with Cyberworks Robotics Inc., Canada. He has published several conferences and journal articles in the fields of autonomous robotics, machine learning, and the Internet of Things (IoT). His current research interests include robotics, self-driving vehicles, artificial intelligence, reinforcement learning, and sensor fusion.

Derek Rayside (Member, IEEE) received the B.A.Sc. degree in systems design engineering and the M.A.Sc. degree in computer engineering from the University of Waterloo and the Ph.D. degree from the Department of Electrical Engineering and Computer Science, MIT. He is currently the Director of software engineering and an Associate Professor with the Electrical and Computer Engineering Department, University of Waterloo, where he is also a Faculty Advisor (along with William Melek) with the Autonomous Vehicle Team in the SAE AutoDrive Challenge (Watonomous.ca). He is a member of SAE and ACM and a licensed P.Eng. in Ontario.

William Melek (Senior Member, IEEE) received the Ph.D. degree in mechanical engineering from the University of Toronto in 2002. He led the Artificial Intelligence Division, Alpha Laboratories Inc. He has founded the Laboratory of Computational Intelligence and Automation, University of Waterloo, in 2004, where he is currently the Director of mechatronics engineering with the RoboHub. He is an expert on robotics, artificial intelligence, sensing, and state estimation. He developed Canada's first industry-ready modular reconfigurable robot (MMR), the state-of-the-art open architecture system is now used in the automotive sector. He has also led the way in designing practical, intelligent, and adaptive control architectures for MMRs based on neural networks. Conceptual prototypes have been developed for the nuclear industry in the USA. He holds 12 Canadian and U.S. patents and his contributions to the manufacturing industry have been featured in the National Post, Globe and Mail, and CBC Television. He was awarded the Young Engineer Medal of Professional Engineers Ontario in 2006. He is the past President of the North American Fuzzy Information Processing Society (NAFIPS).