

Runtime Safety Assurance for Learning-enabled Control of Autonomous Driving Vehicles

Shengduo Chen¹, Yaowei Sun¹, Dachuan Li^{1,2,*}, Qiang Wang², Qi Hao^{1,2,*} and Joseph Sifakis²

Abstract—Providing safety guarantees for Autonomous Vehicle (AV) systems with machine-learning based controllers remains a challenging issue. In this work, we propose Simplex-Drive, a framework that can achieve runtime safety assurance for machine-learning enabled controllers of AVs. The proposed Simplex-Drive consists of an unverified Deep Reinforcement Learning (DRL)-based advanced controller (AC) that achieves desirable performance in complex scenarios, a Velocity-Obstacle (VO) based baseline safe controller (BC) with provably safety guarantees, and a verified mode management unit that monitors the operation status and switches the control authority between AC and BC based on safety-related conditions. We provide a formal correctness proof of Simplex-Drive and conduct a lane-changing case study in dense traffic scenarios. The simulation experiment results demonstrate that Simplex-Drive can always ensure the operation safety without sacrificing control performance, even if the DRL policy may lead to deviations from the safe status.

I. INTRODUCTION

As higher levels of autonomy and intelligence of Autonomous Vehicles (AV) are demanded, there is a growing trend of utilizing data-driven machine-learning (ML) techniques (e.g, Deep Reinforcement Learning, DRL) in AV systems, due to their model-free flexibility and experience learning capabilities. The machine-learning based approaches can achieve superior performance in the control of systems with complex dynamics in dynamic environments.

However, the integration of ML components in the system control framework poses a significant challenge to the safety verification of AV systems. The black-box nature of ML units and lack of explicit logic explainability, prevent using conventional formal verification methods at the design time. The control policies of ML units are derived from the training dataset, hence their safety assurance are implicitly determined by the training data. However, there are no applicable metrics and approaches for the validation and verification of datasets, and hence data-driven component. As a result, the lack of safety assurance prohibits the massive application of ML techniques in safety-critical AV systems.

To address the above challenge, in this paper we seek to leverage the *runtime assurance* approach and propose

a *Simplex-Drive* framework for the control of AVs. The framework is built upon the basic Simplex architecture [1], [2], where the system keeps track of its status at run time, and if necessary, switches the control authority from an unverified performance controller to a proved safe controller to guarantee the runtime safety. When the safe controller is in control, we further incorporate a verified mechanism to enable Simplex-Drive to switch the authority back to the performance controller if safe states can be recovered, so as to restore control performance. In this manner, the system can incorporate ML-based components without requiring correct-by-construction properties at the design stage, while still ensuring provable safety of the overall composite system at run time. Taking the advantage of the Simplex-Drive architecture, this study focuses on the runtime assurance design of Reinforcement Learning-enabled AV controllers with applications to lane-changing scenarios. The primary contributions of this paper are as follows:

- Development of the novel Simplex-Drive architecture for the control of AVs, which can provide provable runtime safety assurance for control systems containing data-driven components.
- Development of a control module with runtime assurance for lane-changing control of AVs, which consists of an unverified DRL-based performance controller and a verified Optimal Reciprocal Collision Avoidance (ORCA, [3]) based safe controller. We provide formal safety specifications for lane-changing control tasks and a set of formal proofs of the correctness of the overall framework.
- Demonstration of the advantages of Simplex-Drive through comparative evaluations based on a number of simulated lane-changing scenarios. The open-source code of Simplex-Drive is available at: https://github.com/625160928/Safety_RL_VO.

The remainder of the paper is organized as follows: Section II reviews the related work. Details of the proposed Simplex-Drive framework are presented in Section III, followed by experimental results and analyses in Section IV. The paper is concluded in Section V.

II. RELATED WORK

The integration of deep reinforcement learning (DRL) into AV applications have been drawing a lot of attention in recent years. Taking advantages of DRL, AVs can efficiently derive control policies and achieve better performance than traditional methods that rely on explicit system models and pre-designed logic [4] [5] [6] [7] [8] [9]. Specifically, a

This work is supported by the Shenzhen Fundamental Research Program (No: JCYJ20200109141622964)

¹S. Chen, Y. Wei, D. Li, Q. Hao are with Department of Computer Science and Engineering, Southern University of Science and Technology, 518055 Shenzhen, China {11860006, sunyw, lidc3}@mail.sustech.edu.cn, haoq@sustech.edu.cn

²D. Li, Q. Wang Q. Hao and J. Sifakis are with Research Institute for Trustworthy Autonomous Systems, 518055 Shenzhen, China wangq8@sustech.edu.cn, joseph.sifakis@imag.fr

*Corresponding authors: Dachuan Li, Qi Hao. S. Chen, Y. Sun and Q. Wang contributed equally to this work

modified Q function approximator [10] is utilized to generate continuous lateral acceleration actions for AVs from the continuous state space [4]. However, the proposed method has been applied to a simple situation assuming that other vehicles only have the longitudinal actions and never change their lanes. For the longitudinal control, a pre-defined Intelligent Driver Model (IDM) has been developed to generate acceleration actions. The later work [11] also uses a neural network to approximate the Q function which can generate continuous actions for longitudinal control. When a lane change decision is made, the network generates actions to adjust longitudinal distances and then a fifth degree polynomial curve is executed to achieve a complete lane change. To control the AV along both lateral and longitudinal directions at the same time, an Asynchronous Learning strategy (A3C) [12] is developed with a network architecture of three convolutional layers and one fully connected layer to generate control actions for AV in race driving [5]. However, such a network can only handle the discrete action space which may result in poor smoothness in trajectories. Other DRL algorithms have also been applied to control AV for lane-change and lane-merge scenarios [6], [7] and [13]. However, none of these approaches can provide theoretical proofs of the safety assurance of generated actions.

The idea of leveraging runtime assurance to address the safety issue of complex and unverified controllers was initially proposed in [1]. The proposed Simplex architecture has been widely adopted in many safety-critical applications such as flight control systems [14], drone surveillance systems [15], collision avoidance of mobile robots [16] and automatic aircraft taxiing control systems [17]. Such domain applications lead to extensions of Simplex focusing on the design of the safety verification/monitoring mechanism and switching logic. The original Simplex leverages Lyapunov based approaches, which can only be applied to continuous systems. Therefore, [18], [19] utilize a set of hybrid automaton and reachability analysis techniques to design the Simplex switching logic for hybrid systems. However, previous Simplex-based frameworks do not focus on any specific controllers and treat all unverified components as black boxes. Moreover, the lack of a reverse switch mechanism prevents the system from restoring control performance. Recently, the Neural Simplex architecture [20] has been proposed to incorporate RL-based components to the Simplex framework, along with a bi-directional switch logic. Inspired by Neural Simplex, we provide runtime assurance for an AV control framework with a DRL-based performance controller in this study.

III. SIMPLEX-DRIVE FRAMEWORK WITH RUNTIME ASSURANCE

A. Overview of Simplex-Drive framework

The overall Simplex-Drive framework is illustrated in Fig. 1. The framework consists of three major components: namely the **Advanced performance controller (AC)**, the **Baseline safe controller (BC)** and the **Mode management unit (MM)**.

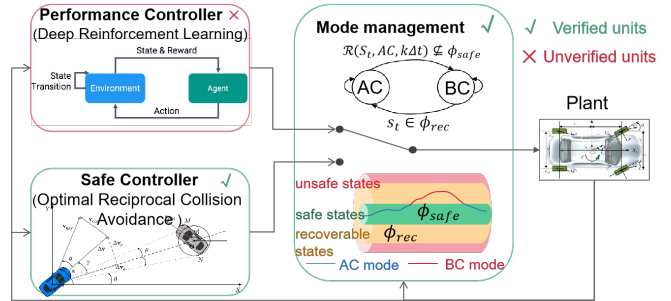


Fig. 1: The system diagram of the proposed Simplex-Drive framework for the machine learning based autonomous vehicle control with safety assurance. AC is used to achieve desired performance and BC for safety assurance. MM monitors the operation condition and switches between these two control authorities.

The dynamics of the controlled vehicle is given by $s_{t+1} = f(s_t, u_t)$, where $s_t \in \mathcal{S}$ and $u_t \in \mathcal{U}$ denote the state and control at time step t , respectively. We denote with $CM_t \in \{AC, BC\}$ as the control mode determined by MM unit at time step t , where AC, BC indicate the AC (BC) controller has the control authority, respectively. Let u_t^{AC}, u_t^{BC} be the control output of AC and BC controller.

The AC is data-driven and maps the current state into a control action: $u_{t+1}^{AC} = \pi_{\theta}^{AC}(s_t)$. In this study, AC can be obtained by training the DRL policy. In contrast, the BC ($u_{t+1}^{BC} = \pi^{BC}(s_t)$) is required to be designed with provable safety guarantee. In the case study of this paper, we design a BC based on the ORCA principle [3], which is proven to preserve collision avoidance property. The MM monitors the state s_t at runtime and switches control authority from AC to BC if u_t^{AC} will lead the system to an *unrecoverable state* (i.e. a state where the system cannot guarantee safety property) at some time step in the future. When $CM_t = BC$, the MM can also switch the mode back to AC to resume AC control if the system returns to a *recoverable state*. Both MM and BC are proven to guarantee safety properties, while AC is not required to be verified correct-by-construction.

B. The Deep Reinforcement Learning Controller

For lane-change scenarios, the process of generating vehicle control commands can be formulated as a Markov Decision Process (MDP) $\langle S, A, P, r, S, \gamma \rangle$. We use the terms “state” to describe the status of the surrounding environment, which can be observed by the agent. Without loss of generality, we assume a continuous action space $a \in A$ for the agent. Let $\pi_{\theta}(a|s)$ be the policy distribution with learnable parameters θ , and $P(s_{t+1}|s_t, a_t)$ the transition probability that measures how likely the environment transitions to s_{t+1} given an action by $a_t \sim \pi_{\theta}(\cdot|s_t)$. After the transition to s_{t+1} , the agent receives a discounted reward $r(s_t, a_t, s_{t+1})$. The $\gamma^i r_{t+i}$ is the discounted return. The objective of the DRL is to solve the above MDP by learning a policy that maximizes the expected total discounted rewards.

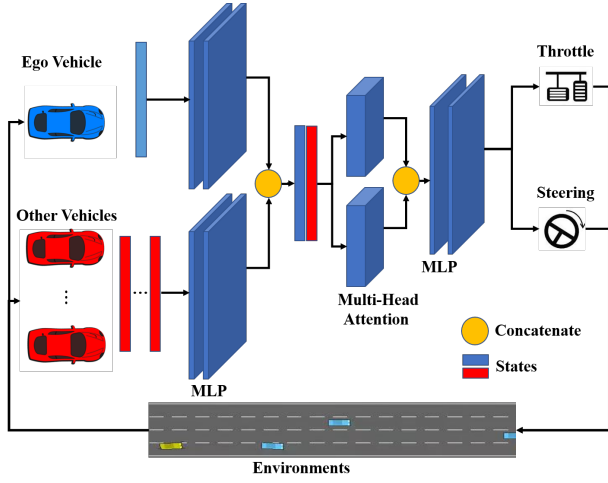


Fig. 2: The network architecture of the proposed deep reinforcement learning based controller. The first two MLP layers are used for environment representation; the multi-head attention layer is used for focusing on nearest vehicles; the last two MLP layers are used for policy formulation.

1) *The Network Architecture*: To design the lane-changing control approach in autonomous driving applications, we develop a model-free policy with a multi-head attention network [21] to map the state to continuous actions. The input states consisting of the position, velocity and heading information of the ego vehicle are fed into two fully-connected layers multi-layer perceptron (MLP) network with 128 neurons and Relu activation functions. The states of surrounding vehicles (which consists of same elements as the ego-vehicle) are sent to another MLP network with the same architecture. Then the concatenated features are sent to a multi-head attention module followed by another MLP with 2 fully connected layers that have 256 neurons. We build the multi-head attention network based on the framework proposed by [21]. The output layer has two units with tanh activation functions that output the throttle and yaw control commands as actions. The overall network architecture is shown in Fig. 2. The policy is trained under the Actor-Critics scheme.

2) *Design of DRL-based Controller*: The reward settings for training the DRL algorithm are set as follows. Reaching the target lane and pose keeping are encouraged by positive rewards, while hazardous behaviors such as collisions and violations of lane borders are penalized by negative rewards. To take the advantage of the DRL-based controller to control the AV more efficiently, we incorporate an additional efficiency-related reward term r_1 and r_3 to encourage the agent to complete lane-changing in a time-efficient manner. The ego vehicle receives r_1 when reaching the target lane and r_2 when collision occurs. r_4 , r_5 are the rewards used to constrain the the vehicle heading.

C. The Optimal Reciprocal Collision Avoidance-based Safe Controller

The Optimal Reciprocal Collision Avoidance (ORCA) [3] approach was originally proposed for multi-robot collision

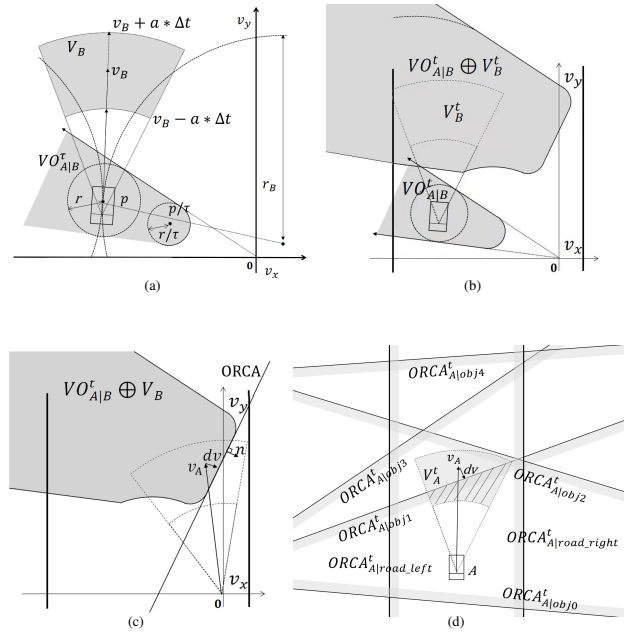


Fig. 3: The Proposed ORCA-Drive safe controller. (a) The reachable velocity set V_B and velocity obstacle set $VO_{A|B}^\tau$ for the ego vehicle A at the origin and the obstacle vehicle B denoted by a rectangle. (b) The resultant velocity obstacle area $VO_{A|B}^t$ for the ego vehicle A at the origin within a period of Δt . The computation of the optimal velocity increment dv for the ego vehicle at the origin given (c) a single obstacle vehicle and (d) multiple obstacle vehicles, respectively.

avoidance and it has been theoretically proved to be able to preserve the safety property. Therefore, we utilize the ORCA as the baseline safe controller in our system to guarantee safety. Since the ORCA is originally designed for omni-directional robot, we extend the ORCA to the control of Ackermann dynamic vehicles. As shown in Fig. 3 (a), surrounding vehicles related to the ego-vehicle movements are regarded as obstacles. Considering the ego vehicle as A with velocity v_A and obstacle as B with v_B , we firstly calculate the velocity reachable set of the obstacle V_B , where the upper bound and lower bounds are calculated with the maximum acceleration a in a predicted time interval Δt . The left and right bounds are calculated with the maximum steering angle and the r_B is the minimum steering radius of the obstacle vehicle with the Ackermann dynamic model. $VO_{A|B}^\tau$ is the velocity obstacle for ego vehicle induced by B for a time window τ with the current relative velocity w.r.t the obstacle vehicle (in a typical AV system configuration, velocities of surrounding vehicles can be measured using onboard sensors such as radars). P is the relative position between the ego and obstacle vehicles; τ is set to 2 in our configuration; r is the radius of the minimum bounding circle of the vehicle.

With the velocity obstacle area $VO_{A|B}^\tau$ at the current relative obstacle vehicle speed, it is able to generated the velocity obstacle area for a period of Δt in the future by

calculating the Minkowski Sum $VO_{A|B}^r \oplus V_B$, as shown in Fig. 3 (b). According to the original ORCA algorithm, the optimal velocity to avoid collision is generated by adding minimal velocity change d_v to the current ego vehicle velocity v_A . In order to address the Ackermann vehicle model constraints, the actions of the ego vehicle should also satisfy the vehicle kinematic constraints. Therefore, we also calculate the velocity reachable set V_A of the ego vehicle. Then the generated velocity $v_{opt} = v_A + d_v$ is checked to make sure it is within V_A . The action generation processes of single and multiple obstacle vehicles are illustrated in Fig. 3 (c) and the Fig. 3 (d) respectively. Details of the extended ORCA (namely ORCA-Drive) algorithm are shown in Algorithm 1.

Algorithm 1 ORCA-Drive Safe Controller

```

1: for Every prediction period  $t$  do
2:   Initialize the ORCA plane list  $L$ 
3:   for each vehicle  $i \in$  other vehicles do
4:     Calculate the velocity reachable set  $V_i^t$  for vehicle  $i$ .
5:     Calculate VO area  $VO_{A|i}^t$  for ego-vehicle  $A$ .
6:     Calculate Minkowski Sum  $CA_{A|i}^t(V_i^t) = V_i^t \oplus VO_{A|i}^t$ .
7:     Calculate minimal velocity change  $dV_i$  that adjust current velocity  $V_A$  to leave  $CA_{A|i}^t(V_i^t)$ .
8:     Generate the ORCA plane  $P_i = [L_i, dV_i]$  and store  $P_i$  in  $L$ .
9:   end for
10:  Generate velocity reachable set  $V_A$  of the ego vehicle under kinematic constraint.
11:  Generate Safe velocity set  $U_s$  of the ego vehicle with ORCA plane list  $L$ .
12:  Calculate the optimal velocity set  $U_{opt} = U_A \cap U_s$  and get optimal velocity change  $dV_{opt}$ .
13:  if  $U_{opt} = \emptyset$  then
14:    for each sampled  $U_n \in U_A$  do
15:      Calculate weight  $W_n += W_{orca} \times$  distance  $d_{orca}$  to ORCA plane for all plane.
16:      Calculate weight  $W_n += W_{current} \times$  distance  $d_{current}$  to current speed.
17:    end for
18:     $V_{opt}$  is the  $U_n$  with highest weight.
19:  else
20:    Get the optimal speed with  $V_{opt} = V_{current} + dV_{opt}$ .
21:  end if
22: end for

```

D. Switching logic

The safety property during operation can be formulated as a safe set: $\phi_{safet} \subseteq \mathcal{S}$, indicating the set of vehicle's states that satisfies safety properties (i.e. safe spacing, traffic rule obedience). At runtime, the system is required to always stay within ϕ_{safe} . We define an additional recoverable region ϕ_{rec} :

Definition. 1 (recoverable region): The recoverable region is the set of states $s \in \phi_{rec}$ that satisfy the following properties:

- (1) $\mathcal{R}(s, AC, \Delta t) \subseteq \phi_{safe}$; (2) $\mathcal{R}(s, BC, \infty) \subseteq \phi_{safe}$; (3) $\mathcal{R}(\mathcal{R}(s, AC, \Delta t), BC, \infty) \subseteq \phi_{safe}$;

where $\mathcal{R}(s_0, MC, \Delta t)$ indicates the reachable sets of the vehicle's states within a time interval Δt under control mode MC , starting from a initial state s_0 . Therefore, ϕ_{rec} indicates a region from which the BC can always steer the system to safe region ϕ_{safe} no matter which controller is in charge.

The switching logic of the MM unit can be modeled as a finite-state automaton with two control locations (Fig.): AC and BC, indicating the AC or BC is in control, respectively. The switching condition is given by:

$$CM_{t+\Delta t} = \begin{cases} BC & CM_t = AC \wedge \mathcal{R}(s_t, AC, k\Delta t) \not\subseteq \phi_{safe} \\ AC & CM_t = BC \wedge s_t \in \phi_{rec} \\ CM_t & otherwise \end{cases} \quad (1)$$

In our application, the MM unit checks the state of the system and makes decisions every Δt . We use bounded reachability analysis [22] to predict the set of vehicle's reachable states and check for deviation from the safe region (line 1, Eq. 1).

E. Runtime Assurance of Simplex-Drive

The correctness of the overall Simplex-Drive can be verified by the following theorem:

Theorem 1: Denoting $\mathcal{R}(s_0)$ as the set of reachable states from s_0 , then $\forall s \in \mathcal{R}(s_0)$, the following invariant holds:

$$(MC = BC \wedge s \in \phi_{safe}) \vee (MC = AC \wedge \mathcal{R}(s, AC, \Delta t) \subseteq \phi_{safe})$$

if the following conditions hold:

- (1) $s_0 \in \phi_{safe}$;
(2) $\Delta t > T_{AC} \wedge \Delta t > T_{BC}$;
(3) $\mathcal{R}(\phi_{safe}, BC, \infty) \subseteq \phi_{safe}$;
(4) $\mathcal{R}(\phi_{rec}, AC, k\Delta t) \subseteq \phi_{safe}, k \in \mathbb{N}, k > 1$

where T_{AC} and T_{BC} denote the control interval of AC and BC.

Proof. Suppose the invariant holds at time step t , the theorem is proved by induction on every consecutive interval of the MM unit:

- (1) If $MC_t = BC$ and there is no mode switch, all future states satisfy the invariant according to condition (3) of Theorem 1;
(2) If $MC_t = BC$ and $BC \rightarrow AC$ at time t , this implies $s_t \in \phi_{rec}$. According to condition (4), $\mathcal{R}(s_t, AC, k\Delta t) \subseteq \phi_{safe}, k > 1$, and the invariant holds for the next interval;
(3) If $MC_t = AC$ and there is no mode switch, this implies $\mathcal{R}(s_t, AC, k\Delta t) \subseteq \phi_{safe} \Rightarrow \mathcal{R}(s_{t+\Delta t}, AC, \Delta t) \subseteq \phi_{safe}$ for the next time interval
(4) If $MC_t = AC$ and $AC \rightarrow BC$ at time t , this implies $\mathcal{R}(s_t, AC, k\Delta t) \not\subseteq \phi_{safe}$. However, as we have $\mathcal{R}(s_{t-\Delta t}, AC, 2\Delta t) \subseteq \phi_{safe}$ (since $MC_t \neq AC$ otherwise) $\Rightarrow \mathcal{R}(s_t, AC, \Delta t) \subseteq \phi_{safe}$. According to condition (2), the BC will execute at least once during the next interval. Therefore, we have $\mathcal{R}(s_t, BC, \infty) \subseteq \phi_{safe}$ according to condition (3) and thus the invariant holds. \square

TABLE I: A PERFORMANCE COMPARISON BETWEEN THE PROPOSED SIMPLEX-DRIVE APPROACH AND OTHER METHODS

Density = 1 (vehicle/lane)	Target Lane Rate	Collision Rate	Avg.Speed (m/s)	Min.Dis (m)	Avg.Min.Dis (m)
ORCA—Drive	90%	0	17.44	19.22	24.80
AVO	63%	35%	18.96	5.35	12.73
Attention PPO	97%	27%	19.44	8.19	16.31
DQN & IDM	100%	36%	19.87	7.55	10.87
Simplex-Drive(Ours)	95%	0	19.36	19.17	21.32
Density = 1.5 (vehicle/lane)	Target Lane Rate	Collision Rate	Avg.Speed (m/s)	Min.Dis (m)	Avg.Min.Dis (m)
ORCA—Drive	77%	0	14.96	10.62	20.52
AVO	33%	72%	18.31	4.74	10.70
Attention PPO	92%	73%	19.44	5.32	12.35
DQN & IDM	100%	95%	19.73	4.55	8.87
Simplex-Drive(Ours)	89%	0	17.91	10.22	19.36
Density = 2 (vehicle/lane)	Target Lane Rate	Collision Rate	Avg.Speed (m/s)	Min.Dis (m)	Avg.Min.Dis (m)
ORCA—Drive	56%	0	15.27	8.30	14.09
AVO	30%	94%	17.66	4.56	8.70
Attention PPO	87%	83%	19.44	5.32	12.35
DQN & IDM	100%	100%	19.76	3.95	6.81
Simplex-Drive(Ours)	85%	0	17.80	7.23	15.78

It is proved in [3] that the ORCA-based BC controller can guarantee safety properties. Therefore, Theorem. 1 guarantees that if the specified conditions hold, all reachable states of the vehicle system always stay within the safe region, no matter which mode the system is currently in.

IV. EXPERIMENT RESULTS

To evaluate the effectiveness and performance of the proposed Simplex-Drive control framework, we conducted extensive experiments in the highway-env simulation environment [23] with different levels of complexity and operation conditions.

A. Experiment Setup

We developed a simulated environment based on highway-env simulator [23] for the training of the DRL-based AC. Various experiments are also conducted using this simulated environment to verify the effectiveness and safety of our framework. In the experiments, we focus on dense traffic scenarios with 3 lanes (the lane width are assumed to be fixed). The controlled ego vehicle (with initial speed V_{ego}) and the obstacle vehicles (with initial speed V_{other}) are assumed to have the same size.

B. Comparative experiments

We compared the proposed architecture with standalone ORCA-Drive controller, AVO controller [24], Attention PPO controller as well as a controller with DQN and IDM [13] in scenarios with different degrees of complexity. The AVO controller utilizes acceleration-velocity obstacle that extend the VO with acceleration constraints to calculate the velocity policy. The DQN with an IDM controller operates within a similar multi-level framework but utilizes DQN to generate high-level decisions and utilize the IDM module for vehicle control. We compared the performance of these approaches in terms of safety and efficiency-related metrics.

The configurations of parameters used in the experiments are shown in TABLE II.

TABLE II: PARAMETER SETTINGS FOR SIMULATIONS

Term	Value	Term	Value
Vehicle Length	5.0m	Vehicle Width	2.0m
V_{max}	20m/s	V_{other}	15m/s
Max_{step}	200	V_{ego}	20m/s
$acceleration_{ego}$	$[-5 m/s^2, 5 m/s^2]$	$Steering_{ego}$	$[-\pi/6, \pi/6]$
λ_1	-0.2	λ_2	0.2
r1	2	r2	-5
r3	$v_{current} - v_{target}$	r4	$cos(\theta)$
V_{target}	20	r5	-2
Lane width	2.5m	Δt	0.5s
W_{orca}	0.7	$W_{current}$	0.3

C. Evaluation Metrics

The evaluation metrics used in the experiments include the target lane rate and collision rate, indicating whether the ego vehicle reaches the target lane before the epoch ends and whether the maneuver results in collisions, respectively. The Avg.Speed is the average vehicle speed at each time step which indicates the efficiency of lane-changing control. In addition, we utilize the minimal distance (Min.Dis) to other vehicles and its average during the epoch (Avg.Min.Dis) to demonstrate the degree of conservativeness of the controllers. We use the vehicle density metric to define the complexity of the environment and the traffic density. The vehicle density is defined as the average number of vehicles on each lane in the near distance. We test the controllers with scenarios with densities of 1, 1.5 and 2. Each density level is tested for 50 times and the results are averaged.

D. Simulation Results

The simulation results are shown in TABLE I. It can be seen that the ORCA-Drive safe controller can always guarantee the safety with 0 collision during all the experiment trials. Compared with DQN and IDM-based controllers that generate continuous control, the Attention PPO controller can

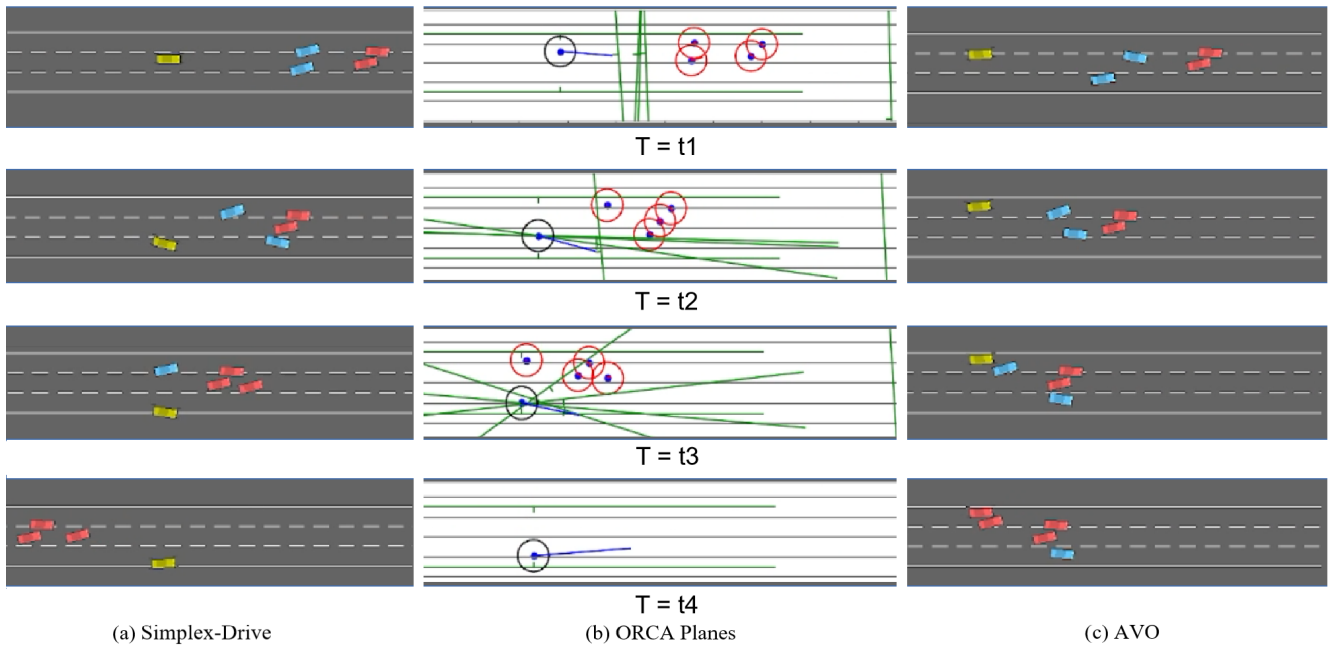


Fig. 4: An illustration of the maneuvers of the ego-vehicle using the Simplex-Drive and ORCA control policies respectively in the presence of traffic accidents. (a) The snap shots of the poses of the ego, crashed, normal vehicles, marked by yellow, red, and blue colors, using the proposed Simplex-Drive controller. (b) ORCA regions of the environment for the Simplex-Drive controller, where the obstacle vehicles and ORCA planes are represented by red circles and green lines respectively. (c) The snap shots of the poses of ego, crashed, and normal vehicles using the AVO controller for the same scenario as a comparison.

deal with the congested environment with better performance and maintain larger minimal spacing to surrounding vehicles. As expected, although the controller with DQN and IDM can achieve smoother trajectories, the lag between the high-level decision module and low-level control module result in delays when reacting to contingencies. The ORCA-Drive safe controller normally keeps a relative larger distance from other vehicles, compared with other controllers, and such a conservative policy generated by ORCA-Drive controller results in a relatively low average speed. In contrast, the proposed Simplex-Drive architecture (with ORCA as BC and attention PPO as AC) can achieve higher success rates while maintaining desirable driving efficiency. In addition, it can also be noticed that the proposed Simplex-Drive control architecture can deal with unexpected cases that do not occur in the DRL training process. Fig. 4 illustrates an example scenario where crashed vehicles block the lane on which the ego-vehicle is driving. In this scenario, the proposed Simplex-Drive controller identifies potential unsafe states and switches to the ORCA-Drive safe control mode and thus avoids collision with crashed vehicles.

To further verify the safety of our switching logic, we conduct another experiment to simulate the circumstance that AC loses control. In such scenario, we use a 'dummy' controller as AC which can only output constant acceleration actions (The ORCA-Drive controller is still used as BC). We evaluate the switches between the controllers and the ratio of the two control modes' (AC and BC) active time among the overall operation time. During the experiment, the ratio of active time of BC increase from 23% to 35%

when 'dummy' controller performed slow acceleration action and from 51% to 58% when 'dummy' controller performed aggressive action. The results show that the ratio of BC controller's active time increases as the the acceleration and traffic density grows, indicating that the switching logic can effectively predict potential unsafe states and take over the AC controller to ensure safety, while avoiding unnecessary frequent switching to maintain the control performance.

V. CONCLUSION

In this paper, we have proposed Simplex-Drive, a reinforcement learning based control framework that can provide runtime safety assurance for autonomous vehicles. We have provided a set of formal proofs for the framework which can preserve safety properties all the time. Within the framework, the RL performance controller can help achieve superior control outputs; the ORCA safe controller can guarantee the drive safe conditions in challenging scenarios; the management unit can achieve smooth switching between two control modes. Simulation experiment results demonstrate that the proposed control system can provide safety guarantee without sacrificing control performance in terms of lane change success rate, lane change speed and trajectory smoothness in typical challenging autonomous driving scenarios. The proposed Simplex-Drive framework can be extended to various data-driven or model-based controllers, and serve as an attempt to develop a more general paradigm for building trustworthy AV systems. Our future work will focus on the implementation and validation of the proposed approach on physical vehicle platforms in real-world scenarios.

REFERENCES

- [1] L. Sha *et al.*, “Using simplicity to control complexity,” *IEEE Software*, vol. 18, no. 4, pp. 20–28, 2001.
- [2] U. Mehmood, S. Bak, S. A. Smolka, and S. D. Stoller, “Safe cps from unsafe controllers,” *arXiv preprint arXiv:2102.12981*, 2021.
- [3] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, “Reciprocal n-body collision avoidance,” in *Robotics research*. Springer, 2011, pp. 3–19.
- [4] P. Wang, C. Chan, and A. D. L. Fortelle, “A reinforcement learning based approach for automated lane change maneuvers,” *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1379–1384, 2018.
- [5] M. Jaritz, R. de Charette, M. Toromanoff, E. Perot, and F. Nashashibi, “End-to-end race driving with deep reinforcement learning,” *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2070–2075, 2018.
- [6] P. Wolf, C. Hubschneider, M. Weber, A. Bauer, J. Härtl, F. Durr, and J. M. Zöllner, “Learning how to drive in a real world simulation with deep q-networks,” *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 244–250, 2017.
- [7] P. Wolf, K. Kurzer, T. Wingert, F. Kuhnt, and J. M. Zöllner, “Adaptive behavior generation for autonomous driving using deep reinforcement learning with compact semantic states,” *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 993–1000, 2018.
- [8] J. Duan, S. E. Li, Y. Guan, Q. Sun, and B. Cheng, “Hierarchical reinforcement learning for self-driving decision-making without reliance on labeled driving data,” *ArXiv*, vol. abs/2001.09816, 2020.
- [9] Y. Chen, C. Dong, P. Palanisamy, P. Mudalige, K. Muelling, and J. Dolan, “Attention-based hierarchical deep reinforcement learning for lane change behaviors in autonomous driving,” *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3697–3703, 2019.
- [10] S. Gu, T. Lillicrap, I. Sutskever, and S. Levine, “Continuous deep q-learning with model-based acceleration,” in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 2829–2838. [Online]. Available: <https://proceedings.mlr.press/v48/gu16.html>
- [11] T. Shi, P. Wang, X. Cheng, C. Chan, and D. Huang, “Driving decision and control for automated lane change behavior based on deep reinforcement learning,” *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 2895–2900, 2019.
- [12] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, “Asynchronous methods for deep reinforcement learning,” in *International conference on machine learning*. PMLR, 2016, pp. 1928–1937.
- [13] M. Toromanoff, É. Wirbel, and F. Moutarde, “End-to-end model-free reinforcement learning for urban driving using implicit affordances,” *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7151–7160, 2020.
- [14] J. D. Schierman, M. D. DeVore, N. D. Richards, N. Gandhi, J. K. Cooper, K. R. Horneman, S. Stoller, and S. Smolka, “Runtime assurance framework development for highly adaptive flight control systems,” Barron Associates, Inc. Charlottesville, Tech. Rep., 2015.
- [15] A. Desai, S. Ghosh, S. A. Seshia, N. Shankar, and A. Tiwari, “Soter: a runtime assurance framework for programming safe robotics systems,” in *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2019, pp. 138–150.
- [16] D. Phan, J. Yang, R. Grosu, S. A. Smolka, and S. D. Stoller, “Collision avoidance for mobile robots with limited sensing and limited information about moving obstacles,” *Formal Methods in System Design*, vol. 51, no. 1, pp. 62–86, 2017.
- [17] D. Cofer, I. Amundson, R. Sattigeri, A. Passi, C. Boggs, E. Smith, L. Gilham, T. Byun, and S. Rayadurgam, “Run-time assurance for learning-based aircraft taxiing,” in *2020 AIAA/IEEE 39th Digital Avionics Systems Conference (DASC)*. IEEE, 2020, pp. 1–9.
- [18] S. Bak, A. Greer, and S. Mitra, “Hybrid cyberphysical system verification with simplex using discrete abstractions,” in *2010 16th IEEE Real-Time and Embedded Technology and Applications Symposium*. IEEE, 2010, pp. 143–152.
- [19] S. Bak, T. T. Johnson, M. Caccamo, and L. Sha, “Real-time reachability for verified simplex design,” in *2014 IEEE Real-Time Systems Symposium*. IEEE, 2014, pp. 138–148.
- [20] D. T. Phan, R. Grosu, N. Jansen, N. Paoletti, S. A. Smolka, and S. D. Stoller, “Neural simplex architecture,” in *NASA Formal Methods Symposium*. Springer, 2020, pp. 97–114.
- [21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in neural information processing systems*, 2017, pp. 5998–6008.
- [22] G. Frehse, C. Le Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler, “Spaceex: Scalable verification of hybrid systems,” in *International Conference on Computer Aided Verification*. Springer, 2011, pp. 379–395.
- [23] E. Leurent, “An environment for autonomous driving decision-making,” <https://github.com/eleurent/highway-env>, 2018.
- [24] J. van den Berg, J. Snape, S. J. Guy, and D. Manocha, “Reciprocal collision avoidance with acceleration-velocity obstacles,” in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 3475–3482.