

Article

Safe Hybrid-Action Reinforcement Learning-Based Decision and Control for Discretionary Lane Change

Ruichen Xu , Jinming Xu , Xiao Liu and Yuan Lin * 

Shien-Ming Wu School of Intelligent Engineering, South China University of Technology, Guangzhou 510641, China; 202220159459@mail.scut.edu.cn (R.X.); wi_jinming@mail.scut.edu.cn (J.X.); 202121060431@mail.scut.edu.cn (X.L.)

* Correspondence: yuanlin@scut.edu.cn

Abstract: Autonomous lane-change, a key feature of advanced driver-assistance systems, can enhance traffic efficiency and reduce the incidence of accidents. However, safe driving of autonomous vehicles remains challenging in complex environments. How to perform safe and appropriate lane change is a popular topic of research in the field of autonomous driving. Currently, few papers consider the safety of reinforcement learning in discretionary lane-change scenarios. We introduce safe hybrid-action reinforcement learning into discretionary lane change for the first time and propose the Parameterized Soft Actor–Critic with PID Lagrangian (PASAC-PIDLag) algorithm. Furthermore, we conduct a comparative analysis with Parameterized Soft Actor–Critic (PASAC), which is an unsafe version of PASAC-PIDLag. Both algorithms are employed to train the lane-change strategy to output both discrete lane-change decisions and continuous longitudinal vehicle acceleration. Our simulation results indicate that at a traffic density of 15 vehicles per kilometer (15 veh/km), the PASAC-PIDLag algorithm exhibits superior safety with a collision rate of 0%, outperforming the PASAC algorithm, which has a collision rate of 1%. The generalization assessments reveal that at low traffic density levels, both the PASAC-PIDLag and PASAC algorithms are proficient in attaining zero collision rates. However, at high traffic density levels, although both algorithms result in collisions, PASAC-PIDLag has a much lower collision rate than PASAC.

Keywords: safe reinforcement learning; lane change; autonomous vehicle; hybrid action spaces



Citation: Xu, R.; Xu, J.; Liu, X.; Lin, Y. Safe Hybrid-Action Reinforcement Learning-Based Decision and Control for Discretionary Lane Change.

Machines **2024**, *12*, 252. <https://doi.org/10.3390/machines12040252>

Received: 1 March 2024

Revised: 9 April 2024

Accepted: 9 April 2024

Published: 11 April 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Nowadays, lane changing is a challenging task that necessitates precise maneuvers to ensure it is conducted safely, comfortably, and swiftly. Lane change include both mandatory and discretionary scenarios [1]. Mandatory lane change refer to the motion planning of lane changing in situations where it is imperative to do so. Scenarios for mandatory lane change include merging from entrance ramps and changing lanes in the presence of obstacles ahead [2]. Discretionary lane change are decisions made by a vehicle to change lanes when it is not demanded due to road conditions, but rather motivated by factors such as speed optimization, driving efficiency, or driver preference. Unlike mandatory lane change, which occur because of immediate necessities such as road obstructions, construction, or merging, discretionary lane change offer an additional layer of complexity to autonomous vehicle algorithms.

Current scholars have provided two research methodologies for decision making in autonomous vehicle lane change: (1) rule-based methods [3–6] and (2) learning-based methods [7,8].

Rule-based decision models use a set of predefined, hand-crafted rules to simulate the decision-making process of drivers. These rules may include adherence to traffic regulations, such as stopping at red lights and proceeding at green lights. The model is highly interpretable because the rules are clear and straightforward, making them easy to

understand and maintain. However, rule-based models may lack flexibility when dealing with complex driving environments and unknown situations because hand-crafted rules may not easily adapt to such complexities and uncertainties [3,6].

Learning-based decision models rely on training models on large-scale driving data to autonomously learn and adapt to different driving conditions. These methods employ deep learning techniques that utilize neural networks and machine learning algorithms to address complex driving decision problems. Although this approach excels in adapting to varied driving scenarios, it has relatively poor interpretability, and there is little guarantee of safety.

Most current learning-based articles are dedicated to using deep reinforcement learning techniques for discretionary autonomous lane-change control of self-driving vehicles [9–14]. In [9], the authors proposed a framework that integrates deep reinforcement learning with Q-masking to enhance the efficiency of autonomous lane change. In [8], the authors enhanced the efficiency of the deep Q-learning algorithm and applied it to the autonomous lane-change scenario [15]. In [10], the authors introduced an automated lane-change method based on reinforcement learning, designing a Q-function approximator with a closed-form greedy policy capable of achieving smooth and efficient driving strategies in various and unpredictable scenarios. In [11], the authors developed a deep reinforcement learning agent capable of robustly executing automated lane change in dynamic and uncertain highway environments, demonstrating superior performance over traditional heuristic-based methods. In [12], the authors applied deep reinforcement learning to address the challenge of successful merging or lane changing for autonomous vehicles in high-density traffic, establishing a benchmark for driving in high-density traffic conditions.

The majority of the literature currently employs discrete reinforcement learning for implementing autonomous lane change [8–14], where the high-level control outputs lane-change decisions using discrete reinforcement learning, and the low-level control uses car-following models such as the Intelligent Driver Model (IDM) [16] to output vehicle acceleration. The decision-making and motion-planning modules, as two closely adjacent and important functional modules of autonomous vehicles, are highly interrelated in terms of functionality and ultimate performance. Therefore, the design of the decision-making process should take into account the feasibility of motion planning. Likewise, motion planning should be formulated based on the decision made [17]. Therefore, in our work, we have adopted a hybrid action space to simultaneously address discrete lane-change decisions and continuous longitudinal acceleration control.

To apply deep reinforcement learning to the autonomous lane-change scenario, ensuring the safety of decision making is essential. There is a paucity of literature considering the safety aspects of autonomous lane change. Given the absence of research using safe reinforcement learning to ensure the safety of discretionary lane change, our paper uses the PID Lagrangian-based hybrid-action reinforcement learning approach [18] to implement autonomous lane change. In [19], the authors proposed a decision-making framework for autonomous vehicles in lane-change scenarios based on deep reinforcement learning with risk awareness. In [20], the authors used a human-driving lane-change decision model combined with regret theory to improve the safety and efficiency of autonomous vehicles in mixed traffic. In [21], the authors introduced a safe reinforcement learning algorithm into the field of autonomous driving, combining the Proximal Policy Optimization (PPO) algorithm with a PID Lagrangian approach to enhance the traffic compliance of motion planners for self-driving vehicles [22].

Safe reinforcement learning [23] is a type of reinforcement learning that incorporates the concepts of safety or risk. Specifically, safe reinforcement learning emphasizes not only pursuing long-term maximum returns during the learning and implementation phases but also adhering to established safety constraints while ensuring reasonable system performance. Compared to Constrained Policy Optimization (CPO) algorithms [24] and safe reinforcement learning algorithms based on Lyapunov functions [25], the Lagrangian-based safe reinforcement learning algorithm performed equally well or even better in

tests within the Safety Gym environment [26]. The oscillations and overshooting observed during the learning process can lead to constraint violations by the agent when applied in practice. Therefore, the PID-based Lagrangian method was proposed [18]. From a control perspective, traditional Lagrange multiplier updates behave as integral control, whereas the PID-based approach introduces proportional and differential controls to stabilize the learning process of the agent.

To the best of our knowledge, there are no existing studies that apply safe hybrid action space algorithms in the domain of discretionary lane change. Previous works have applied hybrid-action reinforcement learning to the discretionary lane-change scenario but have not considered safety [27–30]. In [31], the authors adopted the safe proximal policy optimization algorithm to train the mandatory lane-change policy of an autonomous vehicle. Although the algorithm was designed with safety in mind, the lane-change strategy still exhibited a collision rate of 0.5% in the simulation tests.

The contribution of this paper includes the introduction of a novel safe hybrid-action reinforcement learning algorithm, PASAC-PIDLag, and its application to the discretionary lane-change scenario. We conducted a comprehensive and quantitative comparison between PASAC-PIDLag and its unsafe version PASAC, demonstrating that PASAC-PIDLag outperforms PASAC in terms of both safety and optimality.

Regarding the structure of the paper, Section 2 presents the PASAC-PIDLag and PASAC algorithms, Section 3 discusses the application of the algorithms to lane-change scenarios, Section 4 presents the experiments and results, and Section 5 presents the conclusions.

2. Reinforcement Learning Preliminaries

Reinforcement learning is a computational approach to learning from interaction. In this paradigm, an agent takes actions based on the current state of the environment at each time step. As a result, the environment transitions to another state in the next time step, and the agent receives a reward based on the action taken. Both the actions taken by the agent and the rewards provided by the environment are probabilistic. The goal of an RL algorithm is to maximize the expected discounted cumulative reward.

The framework used to model the environment and the agent's interactions within it in RL is the Markov Decision Process (MDP). An MDP is defined as a tuple (S, A, R, P, γ) , where S is a finite set of states of the environment. A is a finite set of actions that the agent can choose from. P is the state-transition probability matrix. $P(s'|s, a)$ represents the probability of transitioning from state s to state s' after the agent takes action a . R is a reward function. $R(s, a)$ represents the immediate reward the agent receives after taking action a in state s . γ is the discount factor, typically within the range $0 \leq \gamma \leq 1$, which determines the present value of future rewards.

The agent's objective is to discover a policy π , which maps states to the probabilities of selecting each possible action, denoted as $\pi : S \rightarrow A$, that maximizes the expected sum of discounted rewards. The optimal policy π^* can be formally defined as:

$$\pi^* = \arg \max_{\pi} \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \mid s_0 = s, a_t = \pi(s_t) \right], \quad (1)$$

2.1. Soft Actor–Critic

The Soft Actor–Critic (SAC) algorithm [32] is an off-policy, actor–critic reinforcement learning algorithm that incorporates the principles of entropy maximization to balance exploration and exploitation. SAC employs two types of neural networks: soft Q-networks that approximate the soft Q-functions, and policy networks that generate probability distribution over actions. The policy network is trained to maximize the expected reward and entropy. The SAC algorithm optimizes the following entropy-augmented objective function:

$$J(\pi_\theta) = \sum_{t=0}^T \mathbb{E}_{(s_t, a_t) \sim \rho_\pi} [r(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot | s_t))], \quad (2)$$

where π is the policy, θ represents the parameters of the policy, $r(s_t, a_t)$ is the immediate reward for action a_t in state s_t , ρ_π is the distribution over states and actions under policy π , \mathcal{H} is the policy entropy, and α is the entropy coefficient.

SAC uses two Q-networks, Q_{ϕ_1} and Q_{ϕ_2} , to evaluate the policy. The objective of the Q-network, $J_Q(\phi_i)$, is defined as the expected squared error between the current Q-function and the target:

$$J_Q(\phi_i) = \mathbb{E}_{(s, a, r, s') \sim \mathcal{D}} [(Q_{\phi_i}(s, a) - y(r, s', \gamma))^2], \quad (3)$$

$$y(r, s', \gamma) = r + \gamma \left(\min_{i=1,2} Q_{\phi_i}(s', \tilde{a}') - \alpha \log \pi_\theta(\tilde{a}' | s') \right), \quad (4)$$

where \tilde{a}' is the action sampled from the current policy.

To stabilize learning, SAC employs soft target updates to slowly update the target network parameters ϕ'_i :

$$\phi'_i \leftarrow \tau \phi_i + (1 - \tau) \phi'_i, \quad (5)$$

where τ is a small number close to 0, indicating the rate at which the target network parameters are updated.

In the SAC algorithm, actions are selected according to a stochastic policy. This policy is typically parameterized as a Gaussian distribution, allowing the model to capture a range of possible actions. At each timestep t , an action is sampled from this distribution, which is conditioned on the current state s_t :

$$a_t \sim \pi_\theta(\cdot | s_t) = \mathcal{N}(\mu_\theta(s_t), \Sigma_\theta(s_t)), \quad (6)$$

where $\mu_\theta(s_t)$ and $\Sigma_\theta(s_t)$ are the mean and covariance of the policy's Gaussian distribution, respectively, and are functions of the current state s_t parameterized by θ . This stochastic policy approach facilitates exploration of the action space, which is an essential aspect of effective reinforcement learning.

2.2. Parameterized Soft Actor–Critic

Building upon the conventional SAC algorithm, we introduce the Parameterized Soft Actor–Critic (PASAC) algorithm, which is designed to operate within environments that have both discrete and continuous action spaces. In the PASAC algorithm, the policy's output consists of continuous actions along with the probabilities of discrete actions. Let $A_d = \{a_1, a_2, \dots, a_k\}$, where each discrete action $a_i \in A_d$ is associated with a set of continuous parameters $p_{a_i} = \{p_{a_i1}, p_{a_i2}, \dots, p_{a_ik}\} \subseteq \mathbb{R}^k$. Therefore, the action space is represented as $A = \{a_{c1}, a_{c2}, \dots, a_{ck}\} \cup \{a_1, a_2, \dots, a_k\}$, where a_{ck} represents continuous actions, and a_k represents discrete actions.

2.3. Parameterized Soft Actor–Critic with PID Lagrangian

The Constrained Markov Decision Process (CMDP) [33] extends the MDP framework by augmenting it with constraints restricting the set of feasible policies. The CMDP is characterized by the expanded tuple $(S, A, R, P, \gamma, c, d)$, where c is the cost function and d is the corresponding cost limit.

The objective of the CMDP is to optimize policy π , yielding the highest expected sum of discounted rewards over trajectories while keeping the expected sum of discounted costs within the cost limit. Formally, in a CMDP formulation, the RL problem finds the optimal policy π^* that solves

$$\pi^* = \arg \max_{\pi} J_R(\pi) \quad \text{s.t.} \quad J_C(\pi) \leq d, \quad (7)$$

where $J_R(\pi)$ represents the expected reward for the policy π and $J_C(\pi)$ denotes the cost associated with the policy π .

In this study, we address the constrained problem by employing the Lagrangian method, which allows us to convert a constrained problem into an unconstrained problem. Lagrangian techniques are a well-established approach for tackling optimization problems that include constraints. Given the CMDP, the unconstrained problem can be written as

$$\min_{\lambda \geq 0} \max_{\theta} L(\lambda, \theta) = \min_{\lambda \geq 0} \max_{\theta} [J_R(\pi_{\theta}) - \lambda(J_C(\pi_{\theta}) - d)], \quad (8)$$

where L is the Lagrangian and $\lambda \geq 0$ is the Lagrangian multiplier (a penalty coefficient).

In the traditional Lagrangian multiplier method, updates consider only integral control, which is related to the accumulation of constraint violations. Such updates can be conducted within the framework of the Lagrangian method by solving the dual problem, in which the multipliers are adjusted over time to satisfy the constraints.

The Lagrangian multiplier update formula can be represented as

$$\lambda_{k+1} = \max(\lambda_k + \alpha_{\lambda}(J_C(\pi_{\theta}) - d), 0) \quad (9)$$

where α_{λ} is the learning rate of λ .

In the PID method, the dual update rule is enhanced by adding proportional (P) and derivative (D) controls to the existing integral (I) term, with the goal of reducing oscillations in the system output and providing a quicker response to safety constraint violations. The new PID Lagrangian update rule is expressed as

$$\lambda_{\text{new}} = \lambda_{\text{old}} + \left(K_p e(t) + K_i \int e(t) dt + K_d \frac{de(t)}{dt} \right) \quad (10)$$

where $e(t) = J_C(\pi_{\theta}) - d$ is the constraint violation at time t , with d being the target value for the constraint. K_p , K_i , and K_d are the proportional, integral, and derivative gains, respectively. The proportional term $K_p e(t)$ accounts for the current magnitude of the constraint violation, the integral term $K_i \int e(t) dt$ considers the accumulated error over time, and the derivative term $K_d \frac{d}{dt} e(t)$ takes into account the rate of change of the error. This combination helps to satisfy the constraints more quickly and smoothly during the learning process. The pseudocode of the PASAC-PIDLag algorithm is shown in Algorithm 1.

Algorithm 1 Parameterized Soft Actor–Critic with PID Lagrangian

```

1: Algorithm:
2: Initialize  $\theta, \phi_1, \phi_2, \phi'_1 \leftarrow \phi_1, \phi'_2 \leftarrow \phi_2, \mathcal{D}$   $\triangleright$  Init parameters and replay buffer  $\mathcal{D}$ 
3: Initialize PID gains  $K_p, K_i, K_d$ , Lagrangian multiplier  $\lambda$ , target cost  $d$ 
4: Initialize  $J_{c,prev} \leftarrow 0, I \leftarrow 0$   $\triangleright$  Init cost and integral term
5: for each iteration do
6:   for each environment step do
7:      $a_{cont}, a_{disc} \sim \pi_{\theta}(\cdot|s)$ 
8:      $s', r, c \sim \text{Env}(s, a_{cont}, a_{disc})$ 
9:      $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s, a_{cont}, a_{disc}, r, c, s')\}$   $\triangleright$  Store transition
10:  end for
11:  for each gradient step do
12:     $\{s, a_{cont}, a_{disc}, r, c, s'\} \sim \mathcal{D}$   $\triangleright$  Sample batch
13:     $\phi_i \leftarrow \phi_i - \nabla_{\phi_i} J_Q(\phi_i)$   $\triangleright$  Update Q-function parameters
14:     $\theta \leftarrow \theta - \nabla_{\theta} J_{\pi}(\theta)$   $\triangleright$  Update policy parameters
15:     $\phi'_i \leftarrow \tau\phi_i + (1 - \tau)\phi'_i$   $\triangleright$  Update target network parameters
16:     $e \leftarrow J_c - d$ 
17:     $I \leftarrow I + e$ 
18:     $\Delta e \leftarrow J_c - J_{c,prev}$ 
19:     $\lambda \leftarrow \max(\lambda + K_p e + K_i I + K_d \Delta e, 0)$   $\triangleright$  Update  $\lambda$  using PID controller
20:     $J_{c,prev} \leftarrow J_c$ 
21:  end for
22: end for

```

3. Lane-Change Problem Formulation**3.1. Lane-Change Environment**

The lane-change environment was created in the Simulation of Urban Mobility (SUMO) [34] driving simulator. We used a two-lane road with a length of 1 km as our training road, and subsequently, testing was conducted on this road. In this paper, the perception range of the vehicles is represented by a circle with a radius of 200 m, and we assume that the ego vehicle can accurately perceive the status of all vehicles within this range. The surrounding vehicles on the road have an initial speed of 8.33 m/s and a maximum speed of 16.67 m/s, and they use the IDM [16] model for longitudinal control and the SL2015 [35] model for lateral control. In this study, we trained with a traffic flow density of 15 veh/km. As shown in Figure 1, the red vehicle represents the ego vehicle, and the green vehicles represent other vehicles.

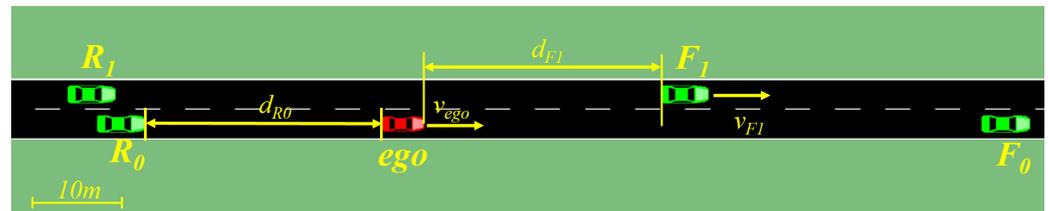


Figure 1. Lane-change environment created using SUMO. v_{ego} is the speed of the *ego* vehicle and v_{F_1} is the speed of the F_1 vehicle. d_{F_1} is the distance between *ego* vehicle and F_1 vehicle, and d_{R_0} is the distance between *ego* vehicle and R_0 vehicle.

3.2. Environment State

In this paper, the state is characterized by ten variables: the distance d_{F_0} between the ego vehicle and the vehicle in front, the distance d_{R_0} between the ego vehicle and the vehicle behind, the distance d_{F_1} between the ego vehicle and the vehicle in front on the target lane, and the distance d_{R_1} between the ego vehicle and the vehicle behind on the target lane. Additionally, the speeds v_{F_0} , v_{F_1} , v_{R_0} , and v_{R_1} of these four vehicles, as well as the speed v_{ego} and acceleration a_{ego} of the ego vehicle, are considered.

$$s = \{v_{F_1}, d_{F_1}, v_{R_1}, d_{R_1}, v_{F_0}, d_{F_0}, v_{R_0}, d_{R_0}, v_{ego}, a_{ego}\}, \quad (11)$$

3.3. Control Action

In this study, the continuous action of the control output is acceleration, and the discrete action is the lane-change decision. Vehicle dynamics and latency are not considered; hence, the vehicle instantaneously executes upon receiving an acceleration command or a lane-change decision. In training, the updates of the vehicle's velocity, position, and lane-change decision occur at a time step of 0.1 s, whereas in testing, the lane-change decision is output every 1 second. Moreover, accounting for the actual vehicle's limits, the limit for continuous actions is defined as $[a_{\min}, a_{\max}] = [-9.8, 5.0] \text{ m/s}^2$, where a_{\min} and a_{\max} represent the minimum and maximum accelerations, respectively.

The action space is defined as a tuple $A = (a_{\text{continuous}}, a_{\text{discrete}})$, where $a_{\text{continuous}}$ represents the continuous control of the vehicle's acceleration, bounded by $a \in [a_{\min}, a_{\max}]$. a_{discrete} is the discrete lane-change decision, where $a_{\text{discrete}} = 1$ indicates changing to another lane, and $a_{\text{discrete}} = 0$ signifies maintaining the current lane.

3.4. Reward

In the context of autonomous vehicle control, reward functions are designed to promote safe, efficient, and comfortable driving behavior. These functions are itemized as follows:

(1) This reward function aims to reduce meaningless lane change caused by the ego vehicle.

$$r_{lc} = \begin{cases} -4, & \text{if } d_{\text{front}} < 25 \text{ m and lane change is decided} \\ -20, & \text{if } d_{\text{front}} \geq 25 \text{ m and lane change is decided} \end{cases} \quad (12)$$

(2) d_{safe} represents the safe following distance from the vehicle ahead in the same lane, which is set to 25 m in this study. v_{limit} denotes the minimum speed limit for the lane when the distance to the vehicle ahead exceeds the safe distance.

$$r_{\text{spd}} = \begin{cases} 0.1 \times |v_{\text{ego}} - v_{\text{limit}}|, & v_{\text{ego}} \in [13.89 \text{ m/s}, 16.67 \text{ m/s}] \text{ and } d_f \geq d_{\text{safe}} \\ -0.1 \times |v_{\text{ego}} - v_{\text{limit}}|, & v_{\text{ego}} \notin [13.89 \text{ m/s}, 16.67 \text{ m/s}] \text{ and } d_f \geq d_{\text{safe}} \end{cases} \quad (13)$$

(3) To facilitate the ego vehicle's acquisition of car-following behavior and to mitigate the risk of collisions, we devised the following reward function predicated on the vehicle-to-vehicle distance metric:

$$r_{\text{dis}} = \begin{cases} -1 \cdot (d_{\text{safe}} - \min(d_{F0}, d_{R0})), & \text{if } d_{F0} \leq d_{\text{safe}} \text{ or } d_{R0} \leq d_{\text{safe}} \\ 0, & \text{otherwise} \end{cases} \quad (14)$$

where d_{R0} represents the distance to the rear vehicle in the same lane and d_{F0} denotes the distance to the forward vehicle in the same lane.

(4) To instruct the ego vehicle to autonomously navigate lane change while mitigating collision occurrences, a penalty of $r_{\text{collision}} = -200$ is incurred following each collision event.

(5) To reduce the jerk during the ego vehicle's motion, we defined the following reward function:

$$r_{\text{jerk}} = -0.005 \times |a_t - a_{t-1}| \quad (15)$$

where a_t represents the acceleration of the ego vehicle at the current time step and a_{t-1} represents the acceleration of the ego vehicle at the previous time step.

(6) For safe reinforcement learning, we employ the TTC as a cost metric. The TTC is expressed as

$$TTC = \frac{v_{ego} - v_{other}}{d_{relative}} \quad (16)$$

where v_{ego} represents the velocity of the ego vehicle, v_{other} denotes the velocity of other vehicles, and $d_{relative}$ indicates the relative distance between the ego vehicle and other vehicles. When the TTC between the ego vehicle and either the leading or following vehicle is less than 2.7 s but greater than 0, the cost is incremented by 1; if the TTC is equal to or greater than 2.7 s or the TTC is not calculable (due to no vehicle being present), the cost remains at 0.

For the PASAC algorithm, the total reward at each timestep is given by

$$r_{total} = r_{lc} + r_{spd} + r_{dis} + r_{jerk} + r_{collision} \quad (17)$$

For the PASAC-PIDLag algorithm, the total reward and cost at each timestep are given by

$$\begin{aligned} r_{total} &= r_{lc} + r_{spd} + r_{dis} + r_{jerk} \\ \text{Cost} &= \begin{cases} \text{Cost} + 1 & \text{if } 0 < \text{TTC} < 2.7 \text{ s} \\ \text{Cost} & \text{if } \text{TTC} \geq 2.7 \end{cases} \end{aligned} \quad (18)$$

We do not include collisions in the cost calculation because the safety policy derived from safe RL may sometimes approach the collision constraint too closely, potentially resulting in collisions.

4. Experiments and Results

In this section, we present the training results under a traffic density of 15 veh/km. The PASAC-PIDLag algorithm outperforms the PASAC-Lag in terms of rewards and costs. The PASAC-Lag method is the traditional Lagrangian method that focuses solely on integral control. Therefore, we did not conduct tests on it. We analyzed both PASAC-PIDLag and PASAC algorithms under a traffic density of 15 veh/km. Additionally, we conducted a generalization analysis of these two algorithms under traffic densities of 10 veh/km and 18 veh/km.

4.1. Training

Our training setup consisted of an NVIDIA RTX 3060 GPU and an Intel i7-12700F CPU, with each training session running for approximately 5 h and covering 400,000 timesteps. The timestep interval was set at 0.1 s to better reflect real-world scenarios. Additionally, we initialized vehicles on the main road within a 50 m buffer zone at the start of each episode. The initial speed of the ego vehicle was set to 8.33 m/s. At the beginning of each episode, the lane for ego vehicle departure was randomly chosen from the two-lane road. During the training process, we evaluated ten episodes for each training episode, and we selected the best-performing policy as the model for subsequent testing.

The hyperparameter configurations for the PASAC-PIDLag, PASAC-Lag, and PASAC algorithms are listed in Table 1. Figure 2 illustrates the training curves for these algorithms. From the training curves, it is evident that the PASAC-PIDLag algorithm demonstrates superior performance compared to both the PASAC-Lag and PASAC algorithms. The training curve of the PASAC-PIDLag algorithm outperforms that of the PASAC-Lag, as the incorporation of PID control in PASAC-PIDLag has successfully reduced the oscillation amplitudes of the cost, leading to more stable performance. Consequently, the PASAC-Lag algorithm was not considered for further testing.

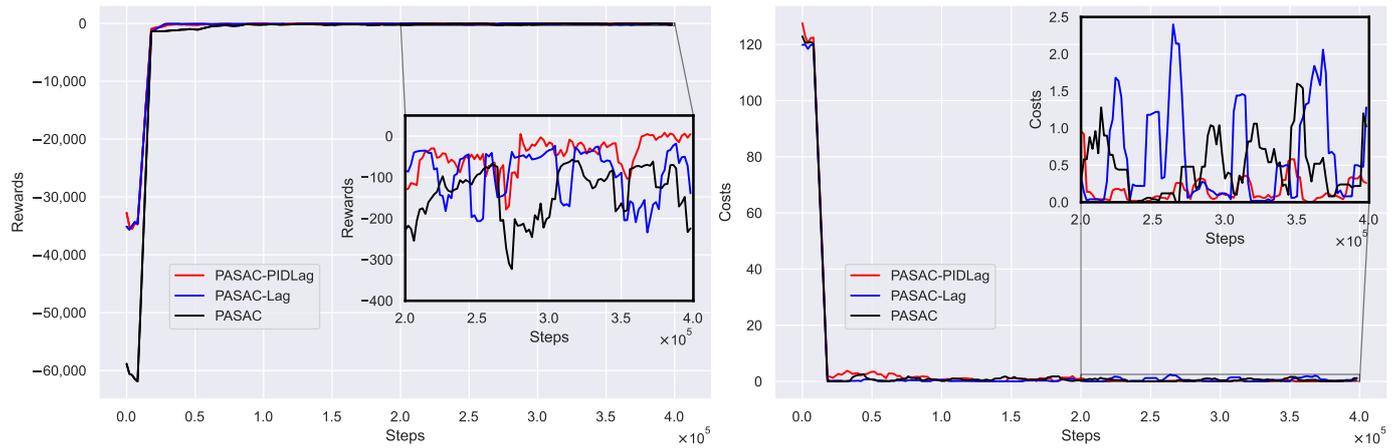


Figure 2. The training progress of the PASAC-PIDLag algorithm compared to the PASAC-Lag and PASAC algorithms.

Table 1. The hyperparameter values for the PASAC-PIDLag, PASAC-Lag, and the PASAC algorithms.

Hyperparameter	PASAC-PIDLag	PASAC-Lag	PASAC
Discount factor γ	0.99	0.99	0.99
Temperature parameter α	0.2	0.2	0.2
The learning rate of the Actor network	0.0001	0.0001	0.0001
The learning rate of the Critic network	0.0003	0.0003	0.0003
Initial learning steps	10,000	10,000	10,000
The size of the batch	256	256	256
The size of the replay buffer	1,000,000	1,000,000	1,000,000
The soft update coefficient	0.005	0.005	0.005
The K_p of the PID controller	0.000002	-	-
The K_i of the PID controller	0.0000002	-	-
The K_d of the PID controller	0.0000001	-	-
Tolerance of constraint violation	0	0	-
Initial Lagrange multiplier value	0.001	0.001	-
Initial value of Lagrangian multiplier	-	0.000002	-

4.2. Testing

In our experiments, we evaluated the performance of the trained policy over 400 episodes under a traffic density of 15 veh/km, encompassing approximately 300,000 timesteps. At the onset of each episode, the initial velocity of the ego vehicle was set to 8.33 m/s (equivalent to 30 km/h). Moreover, to assess the generalizability of our approach, we also conducted tests on the aforementioned strategy at traffic densities of 10 veh/km and 18 veh/km.

4.3. Comparison and Analysis

Based on the results obtained from the dataset of 400 test episodes, as shown in Table 2, it is evident that the PASAC-PIDLag algorithm outperformed the PASAC algorithm on multiple evaluation metrics. The PASAC-PIDLag algorithm exhibited a notably lower collision rate, indicating a safer driving policy adept at mitigating the risk of accidents more effectively. In addition, this algorithm necessitated fewer lane-change maneuvers, suggesting more stable and efficient driving behavior with the potential to diminish disruptive actions within the traffic flow. In terms of velocity, the PASAC-PIDLag algorithm achieved a higher average speed, a pivotal factor in enhancing the rate of transport. Moreover, the jerk metric was significantly reduced for the PASAC-PIDLag algorithm. Upon comprehensive consideration of these performance indicators, the PASAC-PIDLag algorithm surpassed the PASAC algorithm in terms of both optimality and safety.

Table 2. The results under a traffic flow density of 15 veh/km.

400 Episodes	PASAC-PIDLag	PASAC
Average reward	26.35	−53.13
Collision rate	0%	1%
Average speed (m/s)	14.36	14.04
Average jerk (m/s ³)	0.315	0.415
Lane-change times	137	146

Figure 3 depicts an analysis of a lane-changing episode under the PASAC-PIDLag algorithm. Subsequent to this lane-change event, there was an immediate and discernible change in the distance to the preceding vehicle, indicative of the completion of the lane change. The graph detailing relative distance demonstrates that the vehicle initiated the lane-change maneuver when it was at a safe following distance of approximately 25 m. Moreover, the velocity graph depicts a modest escalation in the ego vehicle’s speed following the lane change, which was shortly followed by a decrease. Figure 4 presents the SUMO scene of the successful lane-change maneuver executed by the PASAC-PIDLag algorithm.

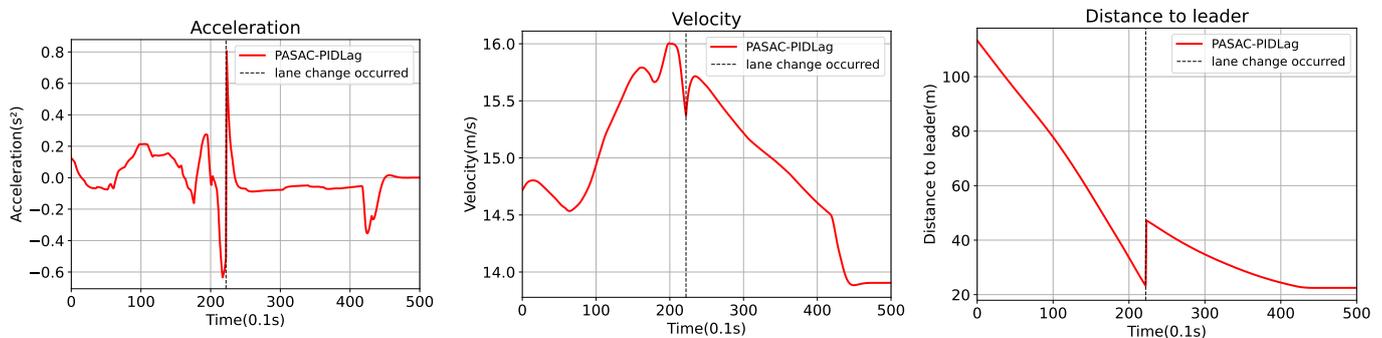


Figure 3. The velocity, acceleration, and distance of the lead vehicle under the regulation of PASAC-PIDLag algorithmic controls. A black dashed line traverses the graphs, symbolizing the execution of a successful lane change by the ego vehicle.

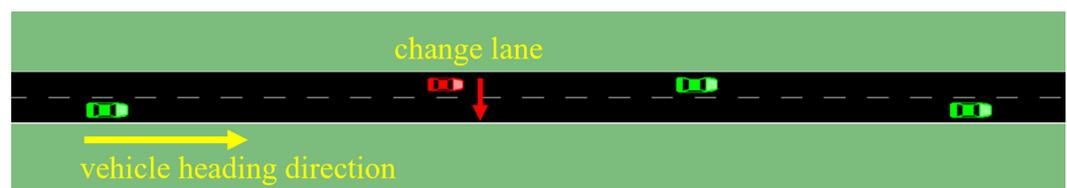


Figure 4. The figure illustrates the successful lane-change maneuver executed by a vehicle under the control of the PASAC-PIDLag algorithm in SUMO, where the red vehicle is denoted as the ego car and the green vehicles represent the surrounding traffic.

Figure 5 depicts an episode of collision occurrence within the PASAC algorithm framework, in which the ego vehicle collided after executing a lane change. The data presented in the figure reveal that the ego vehicle was steadily closing in on the vehicle ahead until the following distance diminished to 19 m, which triggered a decision to change lanes. At this juncture, the presence of another vehicle in the target lane led to a collision. Figure 6 displays the instance of a lane-change maneuver resulting in a collision in SUMO, as directed by the PASAC algorithm.

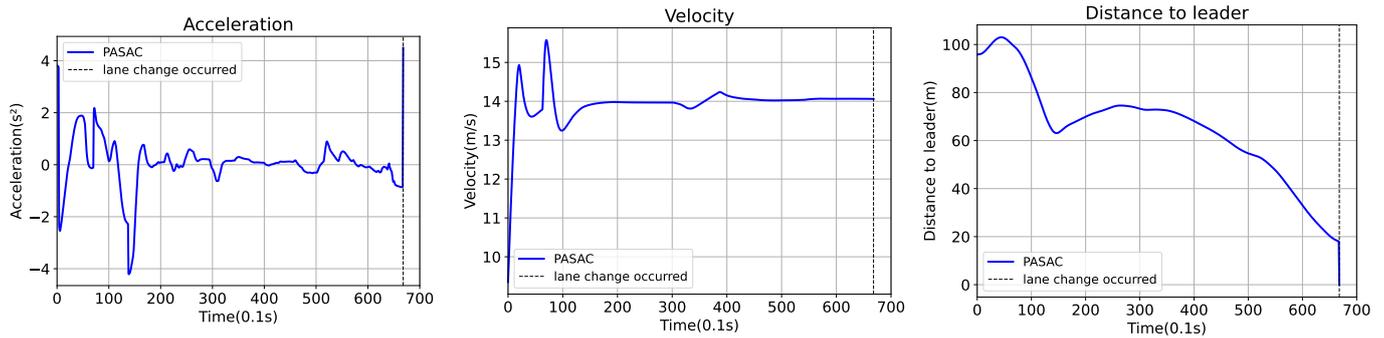


Figure 5. The velocity, acceleration, and distance of the lead vehicle during a collision event due to lane changing under the PASAC algorithm.

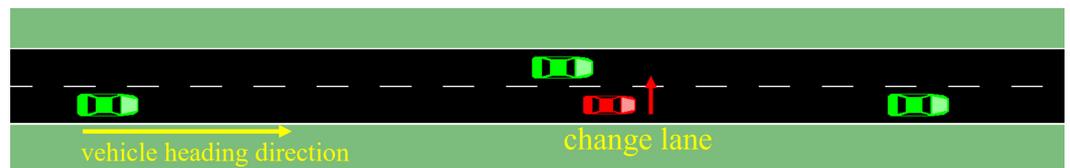


Figure 6. A collision incident during a lane-change maneuver controlled by the PASAC algorithm in SUMO, where the red vehicle represents the ego car and the green vehicle represents other traffic participants.

Figure 7 illustrates another scenario in which a collision occurred under the PASAC algorithm, where the ego vehicle collided during the car-following process. The data and the figure show that due to the presence of a vehicle in the adjacent lane, the ego vehicle was unable to change lanes, resulting in a collision during car following. Figure 8 presents an example of a collision involving an ego vehicle trained using the PASAC algorithm in a car-following scenario in SUMO.

A comparison of lane-changing decisions between the PASAC and PASAC-PIDLag algorithms demonstrated that the strategy derived from the PASAC algorithm was sometimes incapable of effectively balancing the decision related to lane changing and car following under certain conditions.

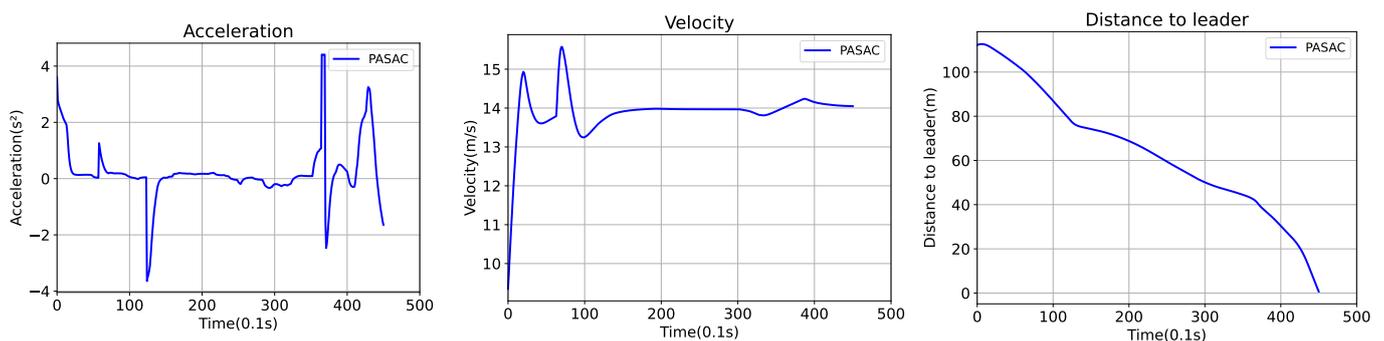


Figure 7. The velocity, acceleration, and distance of the lead vehicle during a collision event due to accelerating under the PASAC algorithm.

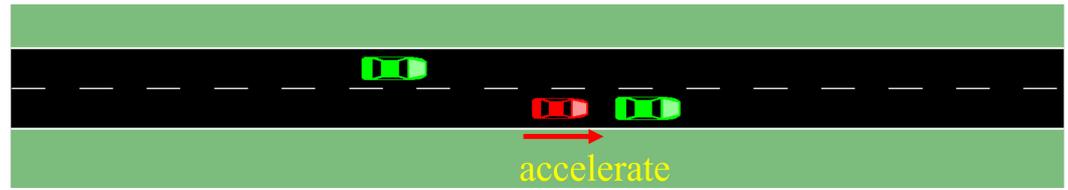


Figure 8. A collision incident during a car-following scenario controlled by the PASAC algorithm in SUMO, where the red vehicle represents the ego car and the green vehicle represents other traffic participants.

4.4. Generalization Analysis

To evaluate the generalizability of the proposed algorithm, we first conducted tests under a traffic density of 10 veh/km, and the results are presented in Table 3. The data presented in Table 3 reveal that at such a reduced traffic density, both algorithms demonstrated the ability to maintain a collision rate of zero. Notwithstanding this equivalence in safety, the PASAC-PIDLag algorithm surpassed its counterpart, PASAC, by securing a greater average reward, attaining a higher mean velocity, and exhibiting a lower average jerk. These findings imply that the PASAC-PIDLag algorithm not only meets safety benchmarks but also excels in performance, offering an enhanced level of optimality over the PASAC algorithm.

Table 3. The generalization results under a traffic flow density of 10 veh/km.

400 Episodes	PASAC-PIDLag	PASAC
Average reward	−24.98	−71.32
Collision rate	0%	0%
Average speed (m/s)	14.51	14.06
Average jerk (m/s ³)	0.290	0.329
Lane-change times	82	78

Our final series of tests were conducted at a traffic flow density of 18 veh/km. The results outlined in Table 4 reveal that at this higher traffic density, the collision rate of the PASAC-PIDLag algorithm remained lower than that of the PASAC algorithm. Furthermore, the PASAC-PIDLag algorithm demonstrated its superiority across all measured metrics, including average reward, average speed, and average jerk.

Table 4. The generalization results under a traffic flow density of 18 veh/km.

400 Episodes	PASAC-PIDLag	PASAC
Average reward	−136.07	−150.25
Collision rate	0.25%	2.25%
Average speed (m/s)	14.17	13.98
Average jerk (m/s ³)	0.290	0.329
Lane-change times	226	164

5. Conclusions

In this paper, we introduced PASAC-PIDLag, a safe hybrid-action reinforcement learning algorithm specifically applied to the scenario of autonomous lane change. This method represents a novel approach that aims to enhance both safety and optimality in the application of reinforcement learning in the autonomous driving domain. We compared it with its unsafe version, PASAC. Both algorithms were trained and tested under a traffic flow density of 0.15 veh/km and underwent generalization tests at densities of 0.10 veh/km and 0.18 veh/km. The results indicated that at a traffic density of 15 veh/km, the strategy trained by the PASAC-PIDLag algorithm managed to maintain zero collisions, while the collision rate for the PASAC algorithm was observed to be 1%.

The PASAC algorithm was observed to encounter two types of collisions at a density of 15 veh/km. The reward structure in this study involves both lane changing and car following, which may lead to collisions arising from unsuccessful lane-changing or car-following maneuvers.

Both algorithms achieved zero collisions at a lower traffic density of 10 veh/km. At a higher traffic density of 18 veh/km, the collision rate of the PASAC-PIDLag algorithm was lower than that of the PASAC algorithm. Across the three traffic densities, the PASAC-PIDLag algorithm consistently achieved higher average speeds, lower average jerks, and greater average rewards. Overall, the PASAC-PIDLag algorithm showed superior performance with respect to safety and optimality.

In future work, we aim to further the application of safe reinforcement learning-based control in actual vehicles. Applying reinforcement learning to real vehicles presents numerous challenges, particularly regarding varying road conditions. In subsequent efforts, we plan to utilize driving simulation software to create road scenarios with obstacles such as construction zones, potholes, and lane congestion. Training within these simulated environments will address the challenge of adapting to diverse road conditions. Additionally, we will employ meta-reinforcement learning to rapidly adapt to different road conditions.

Author Contributions: Conceptualization, Y.L.; methodology, R.X. and J.X.; formal analysis, R.X. and Y.L.; investigation, R.X. and Y.L.; data curation, R.X. and X.L.; writing—original draft preparation, R.X.; writing—review and editing, Y.L., R.X. and J.X.; supervision, Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Guangzhou Basic and Applied Basic Research Program under grant 2023A04J1688 and in part by the South China University of Technology faculty start-up fund.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data can be obtained upon reasonable request from the first author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Kesting, A.; Treiber, M.; Helbing, D. General lane-changing model MOBIL for car-following models. *Transp. Res. Rec.* **2007**, *1999*, 86–94. [[CrossRef](#)]
2. Hou, Y.; Edara, P.; Sun, C. Modeling mandatory lane changing using Bayes classifier and decision trees. *IEEE Trans. Intell. Transp. Syst.* **2013**, *15*, 647–655. [[CrossRef](#)]
3. Aksjonov, A.; Kyrki, V. Rule-based decision-making system for autonomous vehicles at intersections with mixed traffic environment. In Proceedings of the 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), Indianapolis, IN, USA, 19–22 September 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 660–666.
4. Masuda, S.; Nakamura, H.; Kajitani, K. Rule-based searching for collision test cases of autonomous vehicles simulation. *IET Intell. Transp. Syst.* **2018**, *12*, 1088–1095. [[CrossRef](#)]
5. Schwarting, W.; Alonso-Mora, J.; Pauli, L.; Karaman, S.; Rus, D. Parallel autonomy in automated vehicles: Safe motion generation with minimal intervention. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1928–1935.
6. Nilsson, J.; Brännström, M.; Coelingh, E.; Fredriksson, J. Lane change maneuvers for automated vehicles. *IEEE Trans. Intell. Transp. Syst.* **2016**, *18*, 1087–1096. [[CrossRef](#)]
7. Li, L.; Ota, K.; Dong, M. Humanlike driving: Empirical decision-making system for autonomous vehicles. *IEEE Trans. Veh. Technol.* **2018**, *67*, 6814–6823. [[CrossRef](#)]
8. Xu, X.; Zuo, L.; Li, X.; Qian, L.; Ren, J.; Sun, Z. A reinforcement learning approach to autonomous decision making of intelligent vehicles on highways. *IEEE Trans. Syst. Man Cybern. Syst.* **2018**, *50*, 3884–3897. [[CrossRef](#)]
9. Mukadam, M.; Cosgun, A.; Nakhaei, A.; Fujimura, K. Tactical decision making for lane changing with deep reinforcement learning 2017. Available online: <https://openreview.net/pdf?id=HylddmUAZ> (accessed on 1 February 2024).
10. Wang, P.; Chan, C.Y.; de La Fortelle, A. A reinforcement learning based approach for automated lane change maneuvers. In Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV), Suzhou, China, 26–30 June 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1379–1384.

11. Alizadeh, A.; Moghadam, M.; Bicer, Y.; Ure, N.K.; Yavas, U.; Kurtulus, C. Automated lane change decision making using deep reinforcement learning in dynamic and uncertain highway environment. In Proceedings of the 2019 IEEE intelligent transportation systems conference (ITSC), Auckland, New Zealand, 27–30 October 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1399–1404.
12. Saxena, D.M.; Bae, S.; Nakhaei, A.; Fujimura, K.; Likhachev, M. Driving in dense traffic with model-free reinforcement learning. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 5385–5392.
13. Hoel, C.J.; Wolff, K.; Laine, L. Automated speed and lane change decision making using deep reinforcement learning. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 2148–2155.
14. Wang, G.; Hu, J.; Li, Z.; Li, L. Harmonious lane changing via deep reinforcement learning. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 4642–4650. [[CrossRef](#)]
15. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [[CrossRef](#)] [[PubMed](#)]
16. Treiber, M.; Hennecke, A.; Helbing, D. Congested traffic states in empirical observations and microscopic simulations. *Phys. Rev. E* **2000**, *62*, 1805. [[CrossRef](#)]
17. Hang, P.; Lv, C.; Huang, C.; Cai, J.; Hu, Z.; Xing, Y. An integrated framework of decision making and motion planning for autonomous vehicles considering social behaviors. *IEEE Trans. Veh. Technol.* **2020**, *69*, 14458–14469. [[CrossRef](#)]
18. Stooke, A.; Achiam, J.; Abbeel, P. Responsive safety in reinforcement learning by pid lagrangian methods. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 13–18 July 2020; pp. 9133–9143.
19. Li, G.; Yang, Y.; Li, S.; Qu, X.; Lyu, N.; Li, S.E. Decision making of autonomous vehicles in lane change scenarios: Deep reinforcement learning approaches with risk awareness. *Transp. Res. Part C Emerg. Technol.* **2022**, *134*, 103452. [[CrossRef](#)]
20. Chen, D.; Jiang, L.; Wang, Y.; Li, Z. Autonomous driving using safe reinforcement learning by incorporating a regret-based human lane-changing decision model. In Proceedings of the 2020 American Control Conference (ACC), Denver, CO, USA, 1–3 July 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 4355–4361.
21. Wang, X.; Pillmayer, C.; Althoff, M. Learning to obey traffic rules using constrained policy optimization. In Proceedings of the 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), Macau, China, 8–12 October 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 2415–2421.
22. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv* **2017**, arXiv:1707.06347.
23. Garcia, J.; Fernández, F. A comprehensive survey on safe reinforcement learning. *J. Mach. Learn. Res.* **2015**, *16*, 1437–1480.
24. Achiam, J.; Held, D.; Tamar, A.; Abbeel, P. Constrained policy optimization. In Proceedings of the International Conference on Machine Learning, PMLR, Sydney, Australia, 6–11 August 2017; pp. 22–31.
25. Chow, Y.; Nachum, O.; Duenez-Guzman, E.; Ghavamzadeh, M. A Lyapunov-based approach to safe reinforcement learning. *Adv. Neural Inf. Process. Syst.* **2018**, *31*.
26. Ray, A.; Achiam, J.; Amodei, D. Benchmarking safe exploration in deep reinforcement learning. *arXiv* **2019**, arXiv:1910.01708.
27. Lin, Y.; Liu, X.; Zheng, Z.; Wang, L. Discretionary Lane-Change Decision and Control via Parameterized Soft Actor-Critic for Hybrid Action Space. *arXiv* **2024**, arXiv:2402.15790.
28. Gu, Z.; Yin, Y.; Li, S.E.; Duan, J.; Zhang, F.; Zheng, S.; Yang, R. Integrated eco-driving automation of intelligent vehicles in multi-lane scenario via model-accelerated reinforcement learning. *Transp. Res. Part C Emerg. Technol.* **2022**, *144*, 103863. [[CrossRef](#)]
29. Guo, Q.; Angah, O.; Liu, Z.; Ban, X.J. Hybrid deep reinforcement learning based eco-driving for low-level connected and automated vehicles along signalized corridors. *Transp. Res. Part C Emerg. Technol.* **2021**, *124*, 102980. [[CrossRef](#)]
30. Jiang, X.; Zhang, J.; Li, D. Eco-driving for Electric Connected Vehicles at Signalized Intersections: A Parameterized Reinforcement Learning approach. *arXiv* **2022**, arXiv:2206.12065.
31. Ye, F.; Cheng, X.; Wang, P.; Chan, C.Y.; Zhang, J. Automated lane change strategy using proximal policy optimization-based deep reinforcement learning. In Proceedings of the 2020 IEEE Intelligent Vehicles Symposium (IV), Las Vegas, NV, USA, 19 October–13 November 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1746–1752.
32. Haarnoja, T.; Zhou, A.; Hartikainen, K.; Tucker, G.; Ha, S.; Tan, J.; Kumar, V.; Zhu, H.; Gupta, A.; Abbeel, P.; et al. Soft actor-critic algorithms and applications. *arXiv* **2018**, arXiv:1812.05905.
33. Altman, E. *Constrained Markov Decision Processes*; Routledge: London, UK, 2021.
34. Krajzewicz, D.; Erdmann, J.; Behrisch, M.; Bieker, L. Recent development and applications of SUMO-Simulation of Urban MObility. *Int. J. Adv. Syst. Meas.* **2012**, *5*, 3–4.
35. Behrisch, M.; Weber, M. *Modeling Mobility with Open Data: 2nd SUMO Conference 2014 Berlin, Germany, 15–16 May 2014*; Springer: Berlin/Heidelberg, Germany, 2015.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.