International Journal of Engineering

Journal Homepage: www.ije.ir

Safe Reinforcement Learning by Shielding based Reachable Zonotopes for Autonomous Vehicles

H. Raeesi[a], A. Khosravi[a*], P. Sarhadi[b]

[a] *Department of Electrical and Computer Engineering, Babol Noshirvani University of Technology, Babol, Mazandaran, Iran*
[b] *Department of Engineering and Technology, University of Hertfordshire, Hatfield, Hertfordshire, United Kingdom*

PAPER INFO

Paper history:
Received 01 January 2024
Received in revised form 08 April 2024
Accepted 20 May 2024

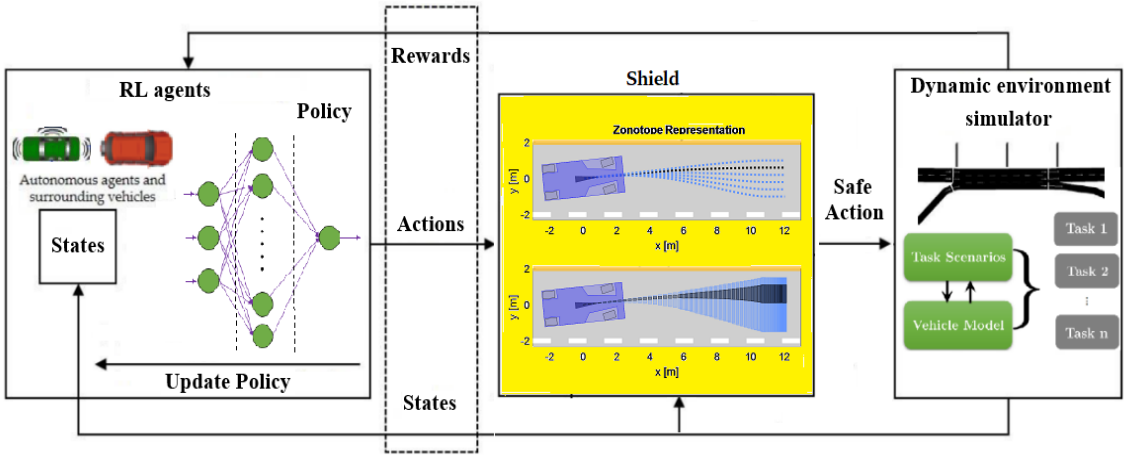
Keywords:
Safe Reinforcement Learning
Shielding
Reachable Set
Autonomous Vehicles

ABSTRACT

The field of autonomous vehicles (AV) has been the subject of extensive research in recent years. It is possible that AVs could contribute greatly to the quality of daily lives if they were implemented. A safe driver model that controls autonomous vehicles is required before this can be accomplished. Reinforcement Learning (RL) is one of the methods suitable for creating these models. In these circumstances, RL agents typically perform random actions during training, which poses a safety risk when driving an AV. To address this issue, shielding has been proposed. By predicting the future state after an action has been taken and determining whether the future state is safe, this shield determines whether the action is safe. For this purpose, reachable zonotopes must be provided, so that at each planning stage, the reachable set of vehicles does not intersect with any obstacles. To this end, we propose a Safe Reinforcement Learning by Shielding-based Reachable Zonotopes (SRLSRZ) approach. It is built around Twin Delayed DDPG (TD3) and compared with it. During training and execution, shielded systems have zero collision. their efficiency is similar to or even better than TD3. A shield-based learning approach is demonstrated to be effective in enabling the agent to learn not to propose unsafe actions. Simulated results indicate that a car vehicle with an unsafe set adjacent to the area that provides the greatest reward performs better when SRLSRZ is used as compared with other methods that are currently considered to be state-of-the-art for achieving safe RL.

doi: *10.5829/ije.2025.38.01a.03*

Graphical Abstract



* Corresponding Author Email: akhosravi@nit.ac.ir (A. Khosravi)

Please cite this article as: Raeesi H, Khosravi A, Sarhadi P. Safe Reinforcement Learning by Shielding based Reachable Zonotopes for Autonomous Vehicles. International Journal of Engineering, Transactions A: Basics. 2025;38(01):21-34.

1. INTRODUCTION

An important aspect of Reinforcement Learning (RL) is its ability to automate decision-making and control. Several recent advancements have been made in challenging research fields, such as robotics, autonomous system control, and games, using algorithms such as Soft Actor-Critic (SAC) (1) and Twin Delayed DDPG (TD3) (2). A notable example of RL in application is the Deep Q-Network (DQN) algorithm (3), a leading algorithm used for decision control in autonomous highway merging (4). Despite its advantages, DQN is primarily suited for discrete decision-making processes. To overcome this, the Deep Deterministic Policy Gradient (DDPG) algorithm has been employed for continuous decision-making in lane changing on single-lane highways (5). However, DDPG too faces challenges like low sample efficiency and unstable network training. In this work To enhance learning efficiency and stability, the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm, which utilizes dual Q-networks and delayed updates, has been introduced (6). The goal of reinforcement learning is to maximize long-term cumulative expected rewards by perceiving consecutive states of an environment and acting accordingly after each observation (7). Nonetheless, for RL agents to be trained and deployed in the real world, safety guarantees are essential. In the absence of these conditions, it is unclear whether the RL agent might cause serious harm to humans the environment, or itself (8, 9). As a result, safe reinforcement learning emerged, which is adapted to ensure that the agent takes safety factors into account in addition to performance during training and operation (10). In this work, we examine RL for guaranteed-safe navigation of autonomous cars in which safety is defined as collision avoidance. RL agents can plan complex sequences of actions, which in combination with reachability analysis is used to ensure safety through postprocessing of the actions.

For many years, RL research has focused on safety (11). Safe RL differs from traditional RL in that it focuses on learning policies that maximize expected rewards on a task without sacrificing safety constraints during both the learning and deployment processes (12). In general, safe RL can be categorized as objective-based and exploration-based (12). The first approach is based on the modification of the optimality criterion to incorporate the concept of risk. It is possible for the agent to explore all actions and states without regard to safety in this instance. As a result, these methods are not safe during training, especially at the beginning, but they eventually tend to converge to safer policies without any guarantees of safety. It has been found that the majority of advances have been made in constrained reinforcement learning (13, 14), in which the policy aims to maximize rewards while meeting user-defined specifications. It is possible to formulate specifications as constraint functions (15, 16) or as temporal logic formulas (17, 18). Second, the exploration process of the learning system is modified so that no unnecessary or catastrophic actions are taken. As a process of exploration, random actions are taken or actions that are not expected to yield maximum rewards (such as greedy strategies) to learn about unexplored states. In spite of this, visiting unexplored states can be harmful to a robot or its environment if it is done naively. In order to avoid this, it may be possible to modify the exploration strategy in order to incorporate risk metrics during both exploration and exploitation, in training as well as in testing (19). In this case, only safe actions are explored in order to achieve only those states that meet the safety specifications. The specification can often be weakened to legal or passive safety in order to achieve provable safety in practice. This means that, in the case of inevitable safety violations caused by other agents, the agent is not responsible for these violations and is therefore viewed as safe (12). A good example of looser restrictions can be found in autonomous driving, where it is sufficient to prove legal safety (20), or in robotics, where a robot hitting a person is assumed to be safe as long as he is at a standstill when the collision occurs (21). The verification process may also utilize an abstraction of the real system, as long as the abstraction is compatible with the real system and covers all relevant aspects of safety (22). Generally, abstractions have a lower complexity than real systems, which facilitates efficient verification. Despite the fact that these approaches are capable of providing strong safety guarantees, the majority of them require prior knowledge of at least some components of the system model (23, 24). The Safe Reinforcement Learning technique used in this study is called Safe Reinforcement Learning by Shielding based Reachable Zonotopes (SRLSRZ). A shield specifically looks at the safety of actions. Its goal is to guarantee safety during training, by only letting the agent perform safe actions. There are a couple of papers which used a shielding type of approach in the field of AVs, although none of them uses the term shielding. Bouton et al. (25) suggested a different approach, in which a probabilistic model computes the probability of reaching the goal state safely for every state-action pair. A threshold is used to determine which of the actions have a high enough chance of success. The agent can then choose one of these actions. The method is tested on an intersection scenario. Krasowski et al. (26) proposed an interesting option and used a system that plans traffic participants' motions. A shield is then used to create a safe subset of actions, based on the other traffic participants' motions, from which the agent can choose. If no safe action exists, a verified fail-safe controller is used. This controller can bring the agent back to a safe state if this is possible. The method was tested on a lane-changing scenario. In a study of collision avoidance, Potential advantage of the

collision avoidance steering compared to the hard braking is estimated (27). In another study, a collision avoidance system that combines the steering and braking inputs is investigated (28). A decision making diagram for the selection of the most effective maneuver among the steering and braking maneuvers by comparing the dimensionless form of the vehicle force required to avoid a collision is presented. The region in which the steering maneuver is superior to the other maneuver is identified (28). Niu et al. (29) proposed a two-stage Safe RL system. In the first stage, a model-free RL algorithm needs to learn to avoid danger at a low speed, while a rule-based shielding type of model checks its actions. In the second stage, the agent needs to learn to drive at a high speed. Now, this rule-based model is replaced with a model based on data, which again acts like a shield. This method is tested in a racing simulator with complex racing tracks. Most of the shields use models which can check whether an action is safe or not, based on different techniques. None of the existing research found in the field of AVs uses a model which can propose safe actions to base a shield on. Considering a set of initial states and a set of possible input signals, SRLSRZ computes the forward reachable set of a vehicle in order to enforce safety (30). In the following step, if there are no intersections between the reachable set and the unsafe set, the system will be verified as safe (31, 32).

There are three contributions to this work. We extend a zonotope representation to move reachable sets away from obstacles. Second, we proposed SRLSRZ as a safe, real-time RL training and deployment system utilizing a continuous action space. Lastly, SRLSRZ is demonstrated on an autonomous car, outperforming baseline RL and other safe RL methods.

The remainder of this paper is organized as follows: Section 2 discusses the modeling of the robot and its environment. In section 3, we compute the robot's reachable sets offline. Section 4 examines the reachable sets online (during training) for safety purposes. Section 5 discusses and evaluates the proposed approach. In section 6, concluding remarks and future directions are discussed.

2. MODELING THE ROBOT AND ENVIRONMENT

2. 1. Vehicle Dynamic Model A standard two-degree of freedom (DOF) single-track (ST) chassis model is shown in Figure 1 (33). In this section, to represent a car driving on a highway, we use the following high-fidelity model adapted from (34).

Paper notation: The index $i \in \{f, r\}$ and $j \in \{l, r\}$ are used to identify vehicle front, rear, left, and right positions, respectively. Table 2 summarizes the parameters used in vehicle dynamics equations and the notations. This model utilizes steady-state assumptions for the lateral dynamics (35). The steady-state slip

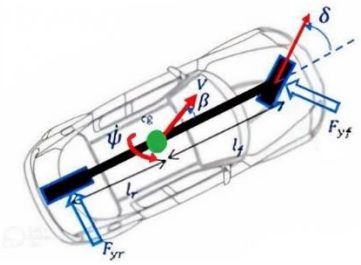


Figure 1. 2-DOF model of lateral vehicle dynamics

angles are related to the road radius as follows. Steady-state force and moment equilibrium equations for the vehicle yield (35):

$$F_{yr} + F_{yf} = \frac{m v_x^2}{R} \quad (1)$$

$$-l_r F_{yr} + l_f F_{yf} = 0 \rightarrow F_{yf} = \frac{l_r}{l_f} F_{yr} \quad (2)$$

From the moment equilibrium Equations 1 and 2, we have:

$$F_{yr} = m \frac{l_f}{L} \frac{v_x^2}{R} \quad , \; F_{yf} = m \frac{l_r}{L} \frac{v_x^2}{R} \quad (3)$$

where $L = l_f + l_r$ is the vehicle length and $m_r = m \frac{l_f}{L}$, $m_f = m \frac{l_r}{L}$ are the portion of the vehicle mass carried on the rear and front axles, respectively.

Assume that the slip angles are small so that the lateral tire force at each wheel is proportional to its slip angle (35):

$$\alpha_f = \frac{F_{yf}}{C_f} = m \frac{l_r}{LC_f} \frac{v_x^2}{R} \quad , \quad \alpha_r = \frac{F_{yr}}{C_r} = m \frac{l_f}{LC_r} \frac{v_x^2}{R} \quad (4)$$

The steady state steering angle is therefore given by Rajamani (35)

$$\delta = \frac{L}{R} + \alpha_f - \alpha_r = \frac{L}{R} + \left(\frac{m l_r}{LC_f} - \frac{m l_f}{LC_r}\right) \frac{v_x^2}{R} \quad (5)$$

Considering the small steering angle and yaw rate equation (35):

$$\dot{\psi} = \omega = \frac{v_x}{R} = \frac{v_x \tan\delta}{L + \left(\frac{m l_r}{LC_f} - \frac{m l_f}{LC_r}\right) v_x^2} \quad (6)$$

According to the geometry of the speed in the rear tire and assume that the slip angles are small we have:

$$v_{r,y} = v_y - \omega l_r, v_{r,x} = v_x \quad (7)$$

$$\alpha_r \approx \tan\alpha_r = \frac{v_{r,y}}{v_{r,x}} \quad (1)$$

From Equations 4, 7 and 8, we have:

$$v_y = \omega l_r + \frac{v}{R}\left(m \frac{l_f}{LC_r} v_x^2\right) \quad (9)$$

So, vehicle state variables are $x = [X, Y, \psi, v_x, \delta]$ with dynamics:

$$\begin{aligned}\dot{X} &= v_x\cos\psi - v_y\cos\psi\\ \dot{Y} &= v_x\sin\psi - v_y\cos\psi\\ \dot{\psi} &= \omega\\ \dot{v}_x &= c_1 v_x + c_2 u_1\\ \dot{\delta} &= c_3(u_2 - \delta)\end{aligned} \quad (10)$$

State variables represent the longitudinal position, lateral position, heading angle, longitudinal velocity, and steering angle. Control variables are acceleration and steering angle, respectively. $c_i : i = 1, \dots, 3$ are model parameters.

2. 2. The Plan Parameter Space This approach has several advantages over traditional MPC approaches. First, reachable sets can be calculated with less conservatism than any other input. In addition, it simplifies the design of the original tracking controller, since there is no need to follow arbitrary trajectory. Moreover, we can verify the safety of continuous time by optimizing the parameters at runtime.

An autonomous vehicle must have a trajectory generator in order to be used under complex conditions. A number of trajectory generation algorithms have been developed in this field, focusing on the different trade-offs between computational complexity, the agility of possible motions, the ability to specify manoeuvre constraints with greater detail, and the ability to handle complex environments (36). Essentially, there are a number of algorithms that deal with the problem of trajectory generation by decoupling geometric and temporal planning: in the first step, a geometric trajectory is constructed that does not include time information, such as lines (37), polynomials (38), Bezier (39), or splines (40). Secondly, the geometric trajectory is parametrized in time to ensure feasibility with respect to dynamics of autonomous vehicles. The disadvantage of lines is that the path is not continuous and therefore jerky, causing uncomfortable transitions between segments. There are several disadvantages associated with bezier curves, including loss of malleability when increasing the curve degree as well as an increase in computation time (more control points must be evaluated and placed correctly) and this planner depends on global waypoints. One disadvantage of splines is that the solution might not be optimal (from the point of view of road fitness and curvature minimization) because the result emphasizes continuity within the parts rather than malleability to fit road constraints. An adverse effect of polynomials is that the curves are usually of the fourth degree or higher, making it difficult to compute the coefficients to determine the motion state. The coefficients have been determined by Mueller et al. (41). One of our essentials is frequent planning for operations on time. Due to its excessive dimensions, it is usually challenging to do so directly with the vehicle dynamic model. Therefore, in this paper, a parametric model of vehicle trajectory planning is proposed. For this purpose, piecewise polynomials are used (41). However, it is necessary to include a fail-safe maneuver in each desired trajectory created by this parametric model so that the vehicle moves to the proper position. We define k as plan parameter space. Let v_{des} be the desired speed at which the car must reach $t_{des}^{[1]} \in (0, t_f)$ and y_{des} be the desired lateral position that the car must reach at $t_{des}^{[2]} \in (0, t_f)$.

$$k = (v_0, y_0, v_{des}, y_{des}) \quad (11)$$

$p_{plan} = (p_1, p_2)$ is a parametric model of planning for the velocity and lateral position of the vehicle (41).

$$p_1(t,k) = \frac{1}{24}c_1(t,k)t^4 + \frac{1}{6}c_2(t,k)t^3 + v_0 t \quad (12)$$

$$\begin{bmatrix}c_1(t,k)\\ c_2(t,k)\end{bmatrix} = \frac{\Delta v_x(t,k)}{(\tau_1(t))^3}\begin{bmatrix}-12\\ 6\tau_1(t)\end{bmatrix} \quad (13)$$

$$\tau_1(t) = \begin{cases} t_{des}^{[1]} & t \in [0, t_{des}^{[1]})\\ t_f - t_{des}^{[1]} & t \in [t_{des}^{[1]}, t_f]\end{cases} \quad (14)$$

$$\Delta v_x(t,k) = \begin{cases} v_{des} - v_0 & t \in [0, t_{des}^{[1]})\\ -v_{des} & t \in [t_{des}^{[1]}, t_f]\end{cases} \quad (15)$$

$$p_2(t,k) = \frac{1}{120}c_3(t,k)t^5 + \frac{1}{24}c_4(t,k)t^4 + \frac{1}{6}c_5(t,k)t^3 - \Delta v_y(t,k)t \quad (16)$$

$$\begin{bmatrix}c_1(t,k)\\ c_2(t,k)\\ c_3(t,k)\end{bmatrix} = \frac{1}{(\tau_2(t))^5}\begin{bmatrix}720 & -360\tau_2(t)\\ -360\tau_2(t) & 168\tau_2(t)^2\\ 60\tau_2(t)^2 & -24\tau_2(t)^3\end{bmatrix}\begin{bmatrix}\Delta y(t,k)\\ \Delta v_y(t,k)\end{bmatrix} \quad (17)$$

$$\tau_2(t) = \begin{cases} t_{des}^{[2]} & t \in [0, t_{des}^{[2]})\\ t_f - t_{des}^{[2]} & t \in [t_{des}^{[2]}, t_f]\end{cases} \quad (18)$$

$$\Delta y(t,k) = \begin{cases} y_{des} - \Delta v_y(t,k) & t \in [0, t_{des}^{[2]})\\ 0 & t \in [t_{des}^{[2]}, t_f]\end{cases} \quad (19)$$

$$\Delta v_y(t,k) = \begin{cases} -v_0 \sin(y_0) & t \in [0, t_{des}^{[2]})\\ 0 & t \in [t_{des}^{[2]}, t_f]\end{cases} \quad (20)$$

Our planning is Receding-horizon (42), which means that the timing at each stage of the planning is as follows:

$$[t_0, t_f] = [t_0, t_{plan}] + [t_{plan}, t_f] \quad (21)$$

At each planning stage, if a safe trajectory is found before the t_{plan}, the vehicle must follow; otherwise, the

vehicle will continue the planned trajectory. The vehicle's center of mass determines the vehicle's position. However, to avoid obstacles, the total volume of the vehicle be considered, so the Forward Occupancy Map (FO) is defined. The volume of the vehicle is in the specified position, and pow(P) is its power set. The volume of the vehicle is shown in Figure 2.

$$FO: X \rightarrow pow(P) \tag{22}$$

Desired speed changes in the longitudinal axis in the time interval $t_{des}^{[1]}$ to reach the desired value and to reach the zero to include a fail-safe maneuver in the time interval $t_f - t_{des}^{[1]}$ and also in the lateral axis to reach the desired location in the time interval $t_{des}^{[2]}$ shown in Figure 3.

2. 3. Tracking Controller In this section, we discuss the control system that can be used to track the desired trajectory of the vehicle to avoid a collision. The trajectory is determined by the parametric planning model discussed in the previous subsection. The two main aspects of controller design are vehicle modelling and control methodologies. The vehicle model selected should have behavior and dynamics similar to those of the actual vehicle, which is a two-DOF dynamic model as described in subsection A. Methodologies for control should take into account feasibility, complexity, and computation of optimal solutions. There are more sophisticated control methods that can be used, but the controller used is of very high quality. This is due to the fact that our desired paths are parameterized by a compact set of parameters. This allows us to design a really effective feedback controller without much effort.

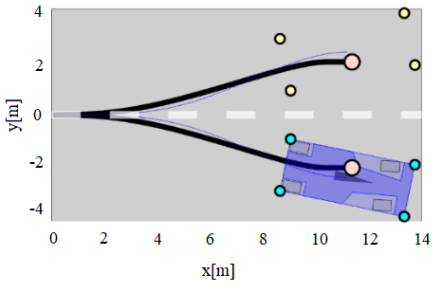


Figure 1. Representation of vehicle volumetric

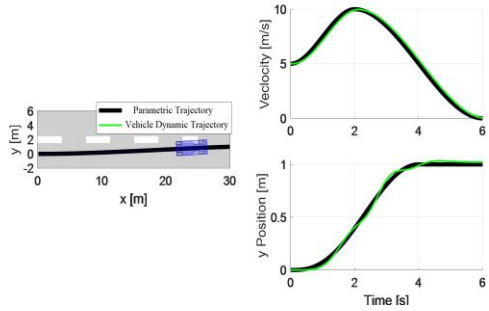


Figure 3. Comparison of the Velocity and lateral Positions of the Vehicle is calculated in two ways: The black colour is obtained from the parametric planning model, and the green colour is calculated from the vehicle dynamic model

2. 3. 1. Pure pursuit Algorithm This process performs by calculating the curvature of the vehicle motion from the current position to the target position. The critical tip of this algorithm is to determine a lookahead point located on the path at a short distance from the vehicle. In this process, the vehicle is thought to be the chaser of this point in a direction a short distance ahead, which explains the algorithm name. We often look a short distance from the front of the vehicle in driving.

As shown in Figure 4, l_{fw} is the anchor point distance from the rear axis, L_{fw} is the distance between the Lookahead point, and η is the angle of the reference path to the Lookahead point. The required steering angle is obtained according to the definitions of the above variables from the following (43).

$$\delta = -\tan^{-1}\left(\frac{L\sin\eta}{\frac{l_{fw}}{2}+l_{fw}\cos\eta}\right) \tag{23}$$

Choosing the Lookahead point is very effective so that if selected too small, the track will be achieved more accurately, but it will create a swinging trajectory. Conversely, if chosen too large, the trajectory fluctuations will be less, but the track will be done less accurately. In this paper, the choice of the Lookahead point is based on the parametric model and does not have a fixed value.

2. 3. 2. Speed Controller The vehicle uses a proportional-derivative controller:

$$u(t, x(t)) = G.\left(\begin{bmatrix} P \\ \psi \\ v_x \\ \delta \end{bmatrix} - \begin{bmatrix} p_{plan}(t,k) \\ 0 \\ \dot{x}(t,k) \\ 0 \end{bmatrix}\right) \tag{24}$$

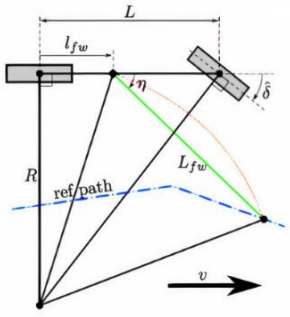


Figure 4. Determining the appropriate steering angle to forward drive in the direction of following the lookahead point in the reference trajectory (43)

where G is a control gains,$P = X \times Y$ and $\dot{x}$ is the time derivative of Equation(12).

2. 4. Presentation of Obstacles A method for displaying the vehicle environment as a limited and discrete set is presented in this section in order to enable the performance of trajectory planning based on reachability analysis to be performed in real time. First, we assume that obstacles are sensed and delivered to us in polygonal form, which makes sense for a sensor like LIDAR. Note that these polygons are not necessarily convex. This assumption holds for common obstacle representations such as occupancy grids or line segments fit to planar point clouds. If an obstacle is not a closed polygon within the sensor horizon (such as a long wall), it can be closed by intersection with the sensor horizon which can be over approximated by a regular polygon. Note that X_{obs} may contain one or more obstacles; the definitions and proofs in this section still hold if it is a union of polygons, which is itself a (potentially disjoint) polygon. Therefore, we refer to X_{obs} as the singular obstacle for ease of exposition (44). The obstacle display method is derived from literature (31), in which all obstacles are buffered in addition to discretization. Using this approach, the selected points will not encounter the parameter space of the trajectory planning algorithm (see Figure 5).

3. OFFLINE REACHABILITY ANALYSIS

3. 1. Calculation of Reachable Set using Planning Model At this point, we have established the high-fidelity and planning models and have begun to relate them through tracking errors. A thorough assessment of diverse trajectory planning methodologies has been conducted by Raeesi et al. (45), with particular attention to address the concurrent challenges of ensuring safety and feasible performance. Additionally, various constraints, including temporal complexity, optimality, completeness, and the requisite model assumptions, have been carefully considered. Ultimately, priority has been given to zonotope-based reachability analysis. We have also established obstacles as portions of the workspace to avoid. To enable the identification of collision-avoiding plans, we define the Reachable set; a relationship is first described as follows:

$$Z_{plan}(\text{Reachable set of parametric model}) + Z_{err} (\text{error rate due to difference between planning and high fidelity model}) = Z_{RS}\ (\text{Reachable set}) \quad (25)$$

The calculation of the reachable set of states is carried out by defining zonotopes. A zonotope is defined as follows (46):

$$\mathcal{Z} := \{c + \sum_{i=1}^{p} \beta_i g^{(i)} \mid \beta_i \in [-1,1]\} \quad (26)$$

Using a more significant number of generators while increasing the number of calculations and complexity will reduce conservatism (Figure 6). Compared to the Sum of Squares (SOS) method, the number of generators used in creating zonotopes with exponential polynomials has the same effect.

Using dynamics, time intervals, and initial condition then produces a set of zonotopes for reachable set of parametric model which (47):

$$Z_{Plan}^{(i)} = c_{plan}^{(i)} + \sum_{n=1}^{n_{RS}} \langle \mathcal{K}^{(i)} \rangle G_{plan}^{(i)} \quad (27)$$

As specified in Figure 7, for each parameter, the planned trajectories are determined, which in Figure 8 is obtained explicitly for one of the states in the selected parameters. The reachable set of parametric models is obtained using zonotopes as shown in the timeframe.

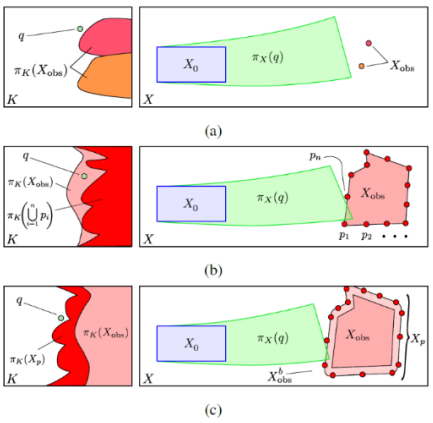


Figure 5. Discretization and buffering of obstacles (X_{obs}) to ensure that the selected point is not encountered by the parametric space of the trajectory planning. The car has footprint X_0 in the xy-subspace X on the right, and the trajectory parameter space K is on the left. The green contour on the right is the reachable set ($\pi_X(q)$) corresponding to the car attempting to track any trajectory (q) from the parameter space (π_K) (31)

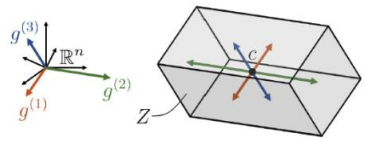


Figure 6. Definition of Zonotope with three generators and a centre (46)

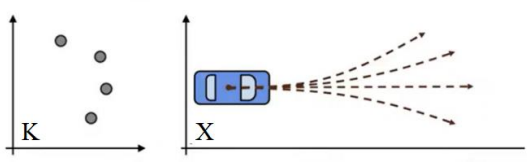


Figure 7. Parameterization of planning trajectories

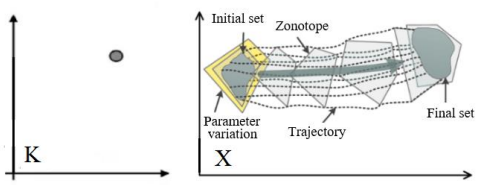


Figure 8. Reachable set of parametric models using zonotopes in planning timeframe

As shown in the figure, based on the decision parameters on the left side of limited reachable space, we will have a shear of the entire planning model space by the reachable set.

While this analysis uses a simple planning model to produce plans, it seeks to compensate for the tracking error caused by the mismatch between the high fidelity and the parametric model of planning (47), which is shown by the zonotope in Figure 9.

This error is defined as a set of zonotope in the following relationship (47):

$$Z_{err}^{(i,j,h)} = c_{err}^{(i,j,h)} + \sum_{n=1}^{\dim(W)} \langle \beta^{(n)} \rangle g_{err}^{(i,j,h)} \quad (28)$$

On the assumption that the amplitude is continuous and there is an infinite number of points (in Figure 7, four points are assumed) to evaluate, we do random sampling on each iteration and identify the conditions under which the tracking error is maximized. We expect that for a given desired track and possible initial speed range, in the most extreme case, there will be a tracking error when the initial speed is as far away from the desired track speed as possible. This shows us how to select speed samples to maximize tracking error.

4. ONLINE SAFE REINFORCEMENT LEARNING

The purpose of this section is to describe online training and testing with SRLSRZ, where the robot selects only safe plans while learning from unsafe ones. In order to enforce safety, we combine the reachable set of the parametric model with the reachable error set in order to construct a reachable set that contains the motion of the vehicle dynamic model when tracking any plan. The safety of a plan is determined by determining whether a subset of the reachable set corresponds to a plan that does not collide.

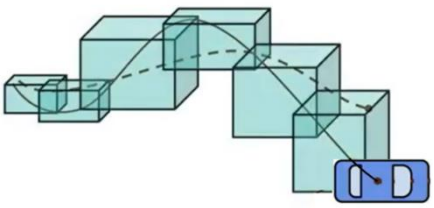


Figure 9. Covering the difference between high fidelity and parametric model of planning by zonotopes

4. 1. POMDP Formulation Considering the fact that the ego vehicle does not have the capability to observe the intentions of surrounding vehicles, it is formulated as a Partially Observable Markov Decision Process (POMDP) for autonomous decision making on the highway. The following is a description of the input state representation, action space, and reward function used to learn the desired driving policy:

4. 1. 1. State Spece We will assume that the environment consists of one agent and several vehicles. Agents are vehicles that perform actions. It is assumed that the vehicle operates in an environment where inter-vehicle communication is not permitted. A vehicle observation consists of six elements. Based on the first four values, the distance between the car's center of mass and the nearest obstacle and the second obstacle can be calculated. It is also possible to observe the vehicle's speed and lateral positions. This allows us to determine the relative position of the road lines, which are defined by the parameters shown in Figure 10.

$$o = (\Delta_{long}^{[1]}, \Delta_{lat}^{[1]}, \Delta_{long}^{[2]}, \Delta_{lat}^{[2]}, v, p_{lat}) \quad (29)$$

4. 1. 2. Action Space It is possible for an agent to choose to go left, stay where it is (i.e. not go left or right), or go right. By influencing steering wheel, gear, and gas pedal inputs, the model will execute the corresponding movement when the agent initiates a lane-change action. There are two common causes of longitudinal and lateral movements: acceleration and deceleration of the agent, and its heading angle as determined by its steering wheel. Therefore, we define the following space of actions:

$$\mathcal{A} = \left\{ \begin{matrix} \text{changing to fast lane, changing to slow lane} \\ \text{driving faster, driving slower} \\ \text{maintaining speed and lane} \end{matrix} \right\}$$

4. 1. 3. Reward Function Reinforcement learning algorithms are primarily based on reward functions. It

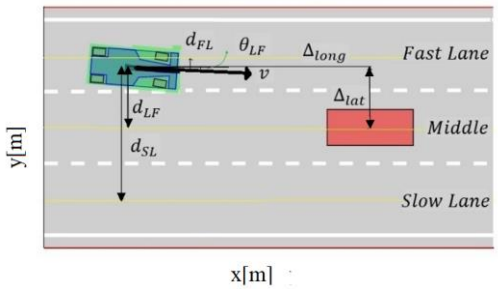


Figure 10. Relative distances from lines and obstacles

must be formulated in a manner that closely resembles the proposed highway driving system. Our primary focus is on safety, efficiency, smoothness, and effort. The following objectives should be achieved by our system:

4. 1. 3. 1. Risk Assessment of Collisions and Near-collisions The penalty for a collision or near-collision is as follows:

$$R_{collision} = \begin{cases} \frac{-1}{\min(D_i)} & \text{if } D_i \text{ lies inside H} \\ -200 & \text{if collision} \end{cases} \quad (30)$$

where $D_i = \sqrt{(x_a - x_i)^2 + (y_a - y_i)^2}$.

As shown in Fgure 11, the hexagonal area (H) surrounding the agent is defined by d_{width}, d_{length}, d_f, and d_b. d_{width} and d_{length} are the width and length of the agent, respectively.

d_b and d_f are safe distances. These distances are not fixed and vary according to changes in the speed of the agent (v_a) and other vehicles ($v_{o,i}$). The definition of d_f is as follows:

$$d_f = v_a^2/2a + v_a/2 + 2d_{length} - v_{o,i}^2/2a \quad (31)$$

This is the minimum distance that the agent should maintain between a vehicle moving in front of it, so that the two vehicles will remain at a distance of d_{length} apart in the event that the front vehicle suddenly stops and the agent brakes after a reaction time of 0.5 seconds. Similarly, d_b is defined in:

$$d_b = v_{o,i}^2/2a + v_{o,i}/2 + 2d_{length} - v_a^2/2a \quad (32)$$

In this case, it relates to vehicles behind the agent. When a collision occurs or when the distance between the agent and other vehicles, which are within the hexagonal area surrounding the agent, decreases, a negative reward is given.

4. 1. 3. 2. Lane following or Changing to Slow Lane and Fast Lane Lane following enables the agent to closely follow the lane. In accordance with d_{LF} and θ_{LF}, moving toward the center of the lane results in a reward, otherwise, a punishment is incurred. A lane change is different from lane following in that it involves decreasing toward a specific lane (fast or slow). In this instance, the lane change has two objectives: 1) Change to the slow lane and 2) Change to the fast lane (Table 1). The agent must achieve fast-lane goals in order to overtake from the slow lane to the fast lane, while slow-lane goals allow the agent to move back to the slow lane once the agent completes the fast-lane goal. A negative reward will be given if the distance between the agent and the lane increases.

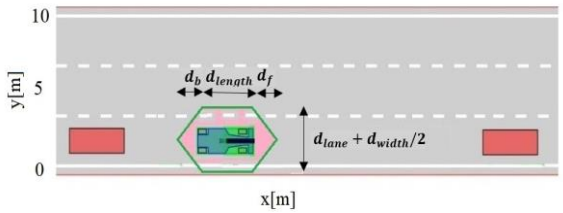


Figure 11. Hexagonal area surrounding the agent for the reinforcement signal calculation

4. 1. 3. 3. Evaluation of Efficiency and Comfort for Target Seeking Target seeking facilitates the agent's ability to reach its target. If the agent moves within line of sight of the target P^*, a higher reward will be awarded. With increasing distance from the target, the reward decreases proportionally. It is important for an agent to be able to move at its maximum possible speed while minimizing acceleration and deceleration and with as little change in heading angle, lateral jerk $\dot{a}_y$ and the longitudinal jerk $\dot{a}_x$ as possible.

$$R_{target\ seeking} = \begin{cases} R_{target\ Heading} = -|\theta_a - \varphi| \\ R_{heading} = -\mu\dot{\theta}_a \\ R_{Jerk} = -\alpha\dot{a}_x - \beta\dot{a}_y \\ R_{speed} = 1 - \frac{v_{desired} - v_a}{v_{desired}} \\ R_{target\ position} = -|P_a - P^*| \end{cases} \quad (33)$$

4. 2. Safe Reinforcement Learning by Shielding based Reachable Zonotopes (SRLSRZ) Usually, an RL agent explores the environment by executing random actions (Figure 12). An agent is placed in a training environment and is required to sample experiences (s, a, r, s′) by executing one action per step. It can learn a policy based on these experiences. Exploration aims to find a wide range of experiences (random action) . This does not work when an AV is trained in real life, since the AV could cause traffic accidents and the equipment might be damaged when it takes random actions. As opposed to exploration, exploitation involves utilizing the knowledge that has already been acquired by the agent. During an exploitation step, the agent picks the (estimated safe action) instead of a random action (Figure 13). A subfield of RL known as Safe RL addresses problems in which the safety of the agent must be ensured. It is an area of RL in which it is crucial to ensure system performance or to respect safety constraints during training and/or execution (12). This research focuses on changing the

TABLE 1. Reward Conditioned on Action Space

Action	R_{action}
Lane Following	$\begin{cases} \gamma & if\ d_{LF} = 0\ \&\vartheta_{LF} = 0 \\ -\lvert d_{LF}\rvert - \lvert\vartheta_{LF}\rvert & otherwise \end{cases}$
Changing to Fast Lane	$-\lvert d_{FL}\rvert$
Changing to Slow Lane	$-\lvert d_{SL}\rvert$

exploration process. Specifically, a method called shielding (48) is analyzed and used to ensure that safety constraints are enforced during the training process. When using shielding, a shield checks whether the actions proposed by the agent are safe or not. If they are not safe, they are overruled with a safe action. This work investigates how shields can be applied in the field of AVs. To do this, a novel type of shield is proposed and tested on the highway scenario.

The SRLSRZ is a kind of teacher that specifically looks at the safety of actions and it has the power to either remove some action options or overrule actions. This method is based on modification of the exploration process.

Kousik (48) and Hunt et al. (49) introduced uniform sampling which is the most common method for exploiting safe actions. It is often necessary to check the safety of every state-action pair when building safe actions online. The use of a single safe action is often appropriate in situations that are time-sensitive and complex. There may be a backup failsafe controller (24) or human feedback (50) that is responsible for this shielding action. An agent can be trained with the unsafe action (s, a, s', r) or with the safe action (s, a_{safe}, s', r), or both when shielding an unsafe action. There is considerable intuition behind both learning tuples. The agent is updated according to its current policy when the original action (s, a, s', r) is selected. Consequently, we can either use the reward provided by the environment $r(s, \tilde{a})$ or penalize the agent for taking an unsafe action by providing a negative reward r^*. When policy is updated based on experience gathered with the most recent policy, learning from the original policy should provide a significant benefit to policy updates.

Therefore, the agent may not be able to learn the underlying dynamics of the system. Instead, we reward the agent for the actual transition performed by using the replacement action tuple(s, a_{safe}, s', r). There is, however, a requirement to update the agent with an action that does not originate from the agent's current policy $\pi(a|s)$. As off-policy learning is expected to behave in this manner, it is assumed that the safe action tuple is more appropriate for off-policy learning than for on-policy learning.

The SRLSRZ proposed in this work consists of two models. For every action, the State Prediction Model predicts the future state after that action. The State Safety Model is then used to determine whether that future state is safe.

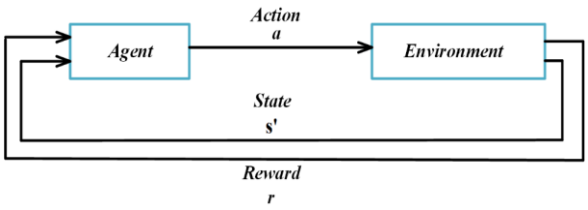


Figure 12. Schematic representation of a Markov Decision Process. An agent sends an action $\boldsymbol{a}$ to the environment, which responds with a reward *r* and transitions to a new state $\boldsymbol{s'}$.

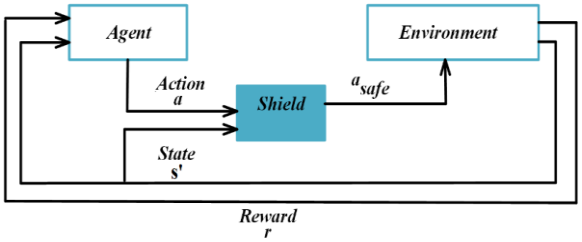


Figure 13. An illustration of a shield during training. At every step, the agent makes a sorted list of actions in descending order of preference based on *s*. The shield picks the first action $\boldsymbol{a_{safe}}$ that is safe from this list.

Algorithm1 Safety Checking Shield

1: $s \leftarrow$ current state
2: $a_i ; i = 1{:}n \leftarrow$ actions based on agent's preference (reward-value)
3: **for** $a \in a_i$ **do**
4: $s_{predicted} \leftarrow$ prediction using plan parameter space by State Prediction Model
5: **if** StateSafetyModel ($s_{predicted}$) **then**
6: **return** a

4. 2. 1. State Prediction Model

The first part of the shielding process is to predict the future state of the environment after one step, based on an action and the current state. The model that makes this prediction is called the State Prediction Model. Based on $\boldsymbol{s_{predict}}$ a second model determines whether the action that resulted in $\boldsymbol{s_{predict}}$ is safe. Therefore it is important that $\boldsymbol{s_{predict}}$ is predicted accurately. State Prediction Model aims to map the current state and an action to an accurate predicted new state. $\boldsymbol{s_{predict}}$ is predicted using Equation 34– and the plan parameter space in section 2.

$$s_{predict} = (x_{predict}, y_{predict}, v_{predict}, d_{predict}, v^{o}_{predict}) \quad (34)$$

Following from the requirement of the State Prediction Model, all variable in $s_{predict}$ should be completely accurate or less safe than the actual future state variables.

4. 2. 2. State Safety Model

The State Safety Model needs to determine whether $\boldsymbol{s_{prdict}}$ is safe or not for every possible. This is basically a function which calculates whether violating safety constraints is unavoidable in $\boldsymbol{s_{predict}}$. This is the case when the safety constraints would be violated for every action that can be taken after reaching $\boldsymbol{s_{predict}}$. To check whether the AV would not hit the vehicle in front of it, the reachable zonotope is required.

By using the Minkowski sum, two zonotopes created in the previous sections are used to determine the available set. According to the operator's property, this set will be in the form of a zonotope.

$$\text{Reachable set} = (Z_{plan} \oplus Z_{err}) \tag{35}$$

The zonotope representation of the reachable set will be as follows (51):

$$\begin{matrix} Z_{plan}^{(i,j)} = Z\left(c_{plan}^{(i,j)}, G_{plan}^{(i,j)}\right) \subset X \times K \\ Z_{err}^{(i,j,h)} = Z\left(c_{err}^{(i,j,h)}, G_{err}^{(i,j,h)}\right) \subset X \end{matrix} \Rightarrow$$
$$Z_{RS}^{(i,j,h)} = z\left(c_{plan}^{(i,j)} + \begin{bmatrix} c_{err}^{(i,j,h)} \\ 0_{n_K \times 1} \end{bmatrix}, \begin{bmatrix} G_{plan}^{(i,j)}, \begin{bmatrix} G_{err}^{(i,j,h)} \\ 0_{n_K \times n_X} \end{bmatrix} \end{bmatrix}\right) \tag{2}$$

Currently, we discuss how to use Reachable zonotopes to create obstacle avoidance restrictions for planning at runtime. The purpose of identifying (conservatively) a $K_{unsf} \subset K$ set contains the plans that can cause encounters. For existing obstacles, a set of zonotopes is defined as $\{Z_{obs}^{m}\}_{m=1}^{n_{obs}}$.

The vehicle must choose a safe $k = (k_{init}, k_{des})$ parameter to plan the trajectory so that:

$$FO(t, x_0, k) \cap \{Z_{obs}^{m}\} = \emptyset \ \forall m \tag{3}$$

The slicing method (51) has been employed to examine the intersection of zonotopes.

Lemma 1: Let X and Y be as in literature (26) Then X and Y intersect if the centre of y is in the zonotope centred at x, with the generators indeterminates of both X and Y (51).

$$X \cap Y \neq \emptyset \Leftrightarrow y \in \left(x + \sum_{i=1}^{r} \langle \chi^{(i)} \rangle g_X^{(i)} + \sum_{j=1}^{s} \langle v^{(j)} \rangle g_Y^{(j)}\right) \tag{38}$$

Notice that this is equivalent to checking if

$$y \in X \oplus \left(0 + \sum_{j=1}^{s} \langle v^{(j)} \rangle g_Y^{(j)}\right) \tag{39}$$

We reorganize the centers and generators of the reachable set and obstacle zonotopes. This lets us leverage the relationship between zonotope intersection and Minkowski sums. The intersection of zonotopes has been checked using the Cora software (52). for more information, you may refer to the related literature (31). In other words, the Minkowski sum enables us to check whether the two zonotopes intersect, which is convenient, as the Minkowski sum of zonotopes is numerically simple (47). Lemma is indicated in Figure 14.

The SRLSRZ that is proposed in this work consists of two models, which together can judge whether an action is safe or not under some given assumptions, depending on the environment. The shielding process is shown in Figure 15 and the pseudocode of it can be found in Algorithm 1. When the two models meet these two requirements, the state prediction model will produce a

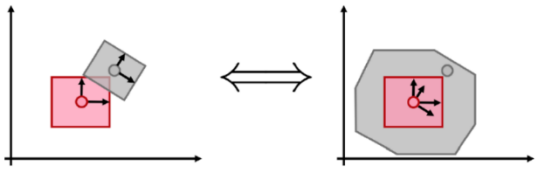


Figure 14. Two zonotopes in pink and grey colours intersect on the left, meaning that the centre of the grey zonotopes is located in the Minkowski sum of pink zonotopes with grey zonotope generators (46)

predicted state which is never safer than the actual future state. The second model will then use this predicted state, which is as unsafe or unsafer than any state that the agent can end up in, and check if there is a safe trajectory, even if everything goes wrong. If a safe trajectory exists, it means that even the unsafest state that the agent can end up in is safe. The agent should never end up in a state where no safe action is available, since actions are only allowed if they put the agent in a state where a safe trajectory is present.

5. RESULTS

To carry out experiments, an Autonomous Vehicle (AV) is needed on which the different proposed systems can be loaded. Therefore, an AV in a simulator is used to evaluate the proposed systems. Multiple driving simulators that can be used for AV research are available From these options; MOBATSim (53) was chosen to use in this research. Simulink 3D Animation with V-Realm is used to visualize driving scenarios in MATLAB 2021. MOBATSim is a simulator that has been specifically developed for training and validation of AVs.

In highway scenario, AV tries to reach a goal position 500 m away on a road-like obstacle course as quickly as possible. The AV is controlled by the RL system until it reaches the end of the road, leaves the road, collides with

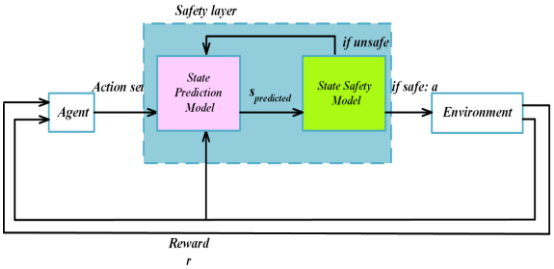


Figure 15. Schematic representation of the safety shielding during training. At every step, the agent sends a sorted list of actions to *the* shield. Based on the current state, the shield predicts the next state for action $\boldsymbol{a}$. If this next state is considered safe by the State Safety Model, $\boldsymbol{a_i}$ is sent to the environment. If it is not considered to be safe, the next action is checked

TABLE 2. Nomenclature and vehicle parameters

Symbols	Value	Unit	Significations
M	1558	kg	Total mass of Vehicle
I_z	2149	kgm^2	Yaw moment of inertia of vehicle
C_f	130000	N/rad	Cornering stiffness of front tiers
C_r	140000	N/rad	Cornering stiffness of rear tiers
L_f	1.46	m	longitudinal distance from c.g.to front tires
L_r	1.41	m	longitudinal distance from c.g.to rear tires
β	-	rad	Side angle at vehicle c.g. (center of gravity)
F_{yf}, F_{yr}	-	N	Lateral tire force on front and rear tires,respectively
F_{xt}	-	N	Total Longitudinal forces
α_f, α_r	-	rad	Side angles at Front and rear tires, Respectively
δ	-	rad	Steering angle
v_x, v_y	-	m/s	Longitudinal and lateral velocity at c.g.of vehicle,respectively
V	[15-30]	m/s	Vehicle velocity interval
R	-	m	Radius of the turn
$\dot{\psi}$	-	rad/s	Yaw rate

another vehicle, or runs out of episode time. The AV is then removed from the road and spawned again at the start of the road. This process repeats itself until the final episode. Screenshots of the highway scenario are shown in Figure .

5. 1. Hyperparameter for Learning Algorithm Our next step is to specify hyperparameters for the learning algorithm (see Table 3 for TD3) that are different from those in Stable Baselines3 (54).

5. 2. Setup A self-driving vehicle is tested on a road-like obstacle course to reach a goal position 500 meters away in the shortest amount of time possible. In our simulation, we used a realistic, high-fidelity model with a larger turning radius at high speeds, so the vehicle must slow down in order to avoid obstacles, or stop if there is insufficient space. As a result of empirical evidence, we used TD3 as our RL agent since it outperforms both SAC and DDPG. We trained TD3 (2) agents for 20,000 episodes and evaluated them on 500 episodes. All the experiments were carried out on the same machine for consistency. The specifications of the machine that is used and the versions of the installed software are shown in Table 4.

Figure 16. Screenshots from the MOBATSim scenarios in which the AV needs to learn how to drive

TABLE 1. Parameters used in the TD3 algorithm

Parameter	Value
Learning rate	0.001
Discount factor	0.98
Batch size	256
Memory size	10^6
Hidden layer 1,2 and 3	256 units
Soft update τ	5×10^{-3}
Policy delay	0.2 s
Gaussian smoothing noise σ	0.2
Activation function	RELU

The following will be assessed:

- How does SRLSRZ compare with a vanilla baseline (unsafe) RL agent and other safe RL methods (e.g., Reachability-based Trajectory Safeguard (RTS+RL) (24) and Safe Advantage-based Intervention for Learning with Reinforcement (SAILR) (55)) with regard to rewards and safety?
- Is it feasible to implement SRLSRZ in real time for safety-critical systems?

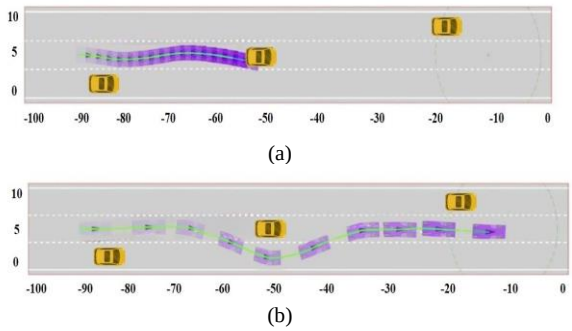
5. 3. Comparison Methods To ensure the safety of autonomous vehicles, we trained for four RL agents: one with SRLSRZ, one with RTS, one with SAILR, and one without any safety layer. The results are summarized in Table 5. In Figure 17, SRLSRZ is shown making two safe lane changes at high speed, while the baseline RL agent is shown colliding. In Figure 18, reward is shown during training. Simulation experiments have demonstrated that shielding layer based on reachable analysis has successfully maneuvered a vehicle at a speed of up to 25 meters per second around a 100-meter test track safely and in real-time, which occurs around other vehicles traveling at a speed of 20 meters per second moving in an unchanged direction. As far as reward and safety are concerned, SRLSRZ outperforms the other methods, is not too conservative, and is capable of operating in real time.

TABLE 4. The specifications of the machine and software versions used in the experiments

Operating system	Microsoft Windows Pro 10
Central Processing Unit	Intel Core i5-11600 CPU @
RAM size	3.9GHz, 6cores
Graphics Processing Unit	32 GB RTX 3060

TABLE 2. Evaluation and comparison results for the car driving experiment

Results	SRLSRZ	Baseline RL	RTS+RL	SAILR
Avg. Planning Time [s]	0.056	**1.4E-5**	0.065	0.030
Goals Reached [%]	**100**	82	88	86
Safety Stopped [%]	**0**	**0**	6	8
Collisions Rate [%]	**0**	12	**0**	**2**
Mean/max Speed [m/s]	19.1/**24.9**	**19.2**/24.8	18.7/**24.9**	**19.2**/**24.9**
Min/Mean/Max Reward	**86**	63	78	68



(a)

(b)

Figure 17. Lane changing in a car with baseline RL (a) and SRLSRZ (b). At each iteration of receding-horizon planning, the car (purple) is plotted. The SRLSRZ agent avoids other vehicles (orange) while traveling at a faster speed than the baseline RL agent, which collides with obstacles

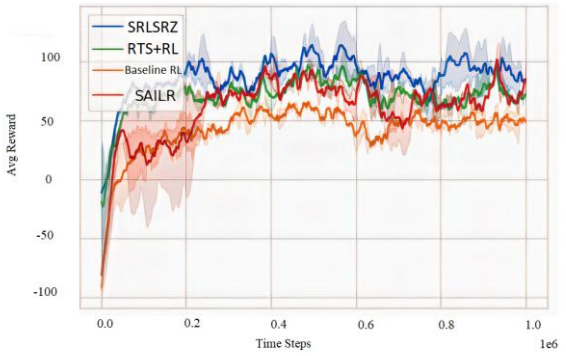


Figure 18. Over time, the average reward for SRLSRZ, other methods and a vanilla TD3 baseline for each experiment

However, SAILR and the TD3 agent achieved higher speeds. Moreover, RTS and SRLSRZ are both safe, whereas SAILR and the baseline RL experience collisions. Our algorithm replans faster than 10 Hz time discretization, making it possible to use it in real time. By driving slowly near obstacles and quickly otherwise, SRLSRZ attains high rewards and goals.

SRLSRZ's performance advantages are explained in the following manner. By automating this high-level behavior, SRLSRZ achieves a higher success rate with less effort on the part of the user (we found that tuning the reward function was easy in practice since there is no need to penalize obstacles/collisions in order to achieve higher success rates). It is possible for SRLSRZ to overcome the sim-to-real gap because it requires much less effort than t_{plan} to ensure safety in each planning iteration. This enables real-time training and evaluation.

6. CONCLUSION

A In this paper, we proposed a Safe Reinforcement Learning by Shielding based Reachable Zonotopes, or SRLSRZ. The method is demonstrated in simulation by performing safe, real-time receding-horizon planning for automobiles in a highway scenario with continuous action spaces . In terms of reward and task completion, SRLSRZ is typically superior to state-of-the-art safe trajectory planners. As part of future research, SRLSRZ will be applied to hardware and non-rigid-body robots, and additional benefits of safe RL training will be explored.

7. REFERENCES

1. Haarnoja T, Zhou A, Abbeel P, Levine S, editors. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. International conference on machine learning; 2018: PMLR. https://doi.org/10.48550/arXiv.1801.01290
2. Fujimoto S, Hoof H, Meger D, editors. Addressing function approximation error in actor-critic methods. International conference on machine learning; 2018: PMLR. https://doi.org/10.48550/arXiv.1802.09477
3. Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, et al. Human-level control through deep reinforcement learning. nature. 2015;518(7540):529-33. 2015;518(7540):529-533. https://doi.org/10.1038/nature14236
4. He X, Lou B, Yang H, Lv C. Robust decision making for autonomous vehicles at highway on-ramps: A constrained adversarial reinforcement learning approach. IEEE Transactions on Intelligent Transportation Systems. 2022;24(4):4103-13. 10.1109/TITS.2022.3229518
5. Hu H, Lu Z, Wang Q, Zheng C. End-to-End automated lane-change maneuvering considering driving style using a deep deterministic policy gradient algorithm. Sensors. 2020;20(18):5443. https://doi.org/10.3390/s20185443
6. Hadi B, Khosravi A, Sarhadi P. Deep reinforcement learning for adaptive path planning and control of an autonomous underwater vehicle. Applied Ocean Research. 2022;129:103326. https://doi.org/10.1016/j.apor.2022.103326
7. Sutton RS, Barto AG. Introduction to reinforcement learning: MIT press Cambridge; 1998. 10.1109/TNN.1998.712192

8. Mihatsch O, Neuneier R. Risk-sensitive reinforcement learning. Machine learning. 2002;49:267-90. https://doi.org/10.1023/A:1017940631555
9. Dalal G, Dvijotham K, Vecerik M, Hester T, Paduraru C, Tassa Y. Safe exploration in continuous action spaces. arXiv preprint arXiv:180108757. 2018. https://doi.org/10.48550/arXiv.1801.08757
10. Schulman J, Levine S, Abbeel P, Jordan M, Moritz P, editors. Trust region policy optimization. International conference on machine learning; 2015: PMLR. https://doi.org/10.48550/arXiv.1502.05477
11. Yaghmaee F, Koohi H. Dynamic obstacle avoidance by distributed algorithm based on reinforcement learning. International Journal of Engineering, Transactions B: Applications. 2015;28(2):198-204. doi: 10.5829/idosi.ije.2015.28.02b.05
12. Garcıa J, Fernández F. A comprehensive survey on safe reinforcement learning. Journal of Machine Learning Research. 2015;16(1):1437-80. https://dl.acm.org/doi/10.5555/2789272.2886795
13. Achiam J, Held D, Tamar A, Abbeel P, editors. Constrained policy optimization. International conference on machine learning; 2017: PMLR. https://doi.org/10.48550/arXiv.1705.10528
14. Stooke A, Achiam J, Abbeel P, editors. Responsive safety in reinforcement learning by pid lagrangian methods. International Conference on Machine Learning; 2020: PMLR. https://doi.org/10.48550/arXiv.2007.03964
15. Marvi Z, Kiumarsi B. Safe reinforcement learning: A control barrier function optimization approach. International Journal of Robust and Nonlinear Control. 2021;31(6):1923-40. https://doi.org/10.1002/rnc.5132
16. Yang T-Y, Rosca J, Narasimhan K, Ramadge PJ. Projection-based constrained policy optimization. arXiv preprint arXiv:201003152. 2020. https://arxiv.org/abs/2010.03152
17. De Giacomo G, Iocchi L, Favorito M, Patrizi F, editors. Foundations for restraining bolts: Reinforcement learning with LTLf/LDLf restraining specifications. Proceedings of the international conference on automated planning and scheduling; 2019. https://doi.org/10.1609/icaps.v29i1.3549
18. Hasanbeig M, Abate A, Kroening D. Cautious reinforcement learning with logical constraints. arXiv preprint arXiv:200212156. 2020. https://doi.org/10.48550/arXiv.2002.12156
19. Gehring C, Precup D, editors. Smart exploration in reinforcement learning using absolute temporal difference errors. Proceedings of the 2013 International Conference on Autonomous agents and multi-agent systems; 2013. https://dl.acm.org/doi/10.5555/2484920.2485084
20. Pek C, Manzinger S, Koschi M, Althoff M. Using online verification to prevent autonomous vehicles from causing accidents. Nature Machine Intelligence. 2020;2(9):518-28. https://doi.org/10.1038/s42256-020-0225-y
21. Bouraine S, Fraichard T, Salhi H, editors. Provably safe navigation for mobile robots with limited field-of-views in unknown dynamic environments. 2012 IEEE International Conference on robotics and automation; 2012: IEEE. DOI:10.1109/ICRA.2012.6224932
22. Roehm H, Oehlerking J, Woehrle M, Althoff M. Model conformance for cyber-physical systems: A survey. ACM Transactions on Cyber-Physical Systems. 2019;3(3):1-26. DOI:10.1145/3306157
23. Lew T, Sharma A, Harrison J, Bylard A, Pavone M. Safe active dynamics learning and control: A sequential exploration–exploitation framework. IEEE Transactions on Robotics. 2022;38(5):2888-907. https://doi.org/10.48550/arXiv.2008.11700
24. Shao YS, Chen C, Kousik S, Vasudevan R. Reachability-based trajectory safeguard (RTS): A safe and fast reinforcement learning safety layer for continuous control. IEEE Robotics and Automation Letters. 2021;6(2):3663-70. https://doi.org/10.48550/arXiv.2011.0842
25. Bouton M, Nakhaei A, Fujimura K, Kochenderfer MJ, editors. Safe reinforcement learning with scene decomposition for navigating complex urban environments. 2019 IEEE Intelligent Vehicles Symposium (IV); 2019: IEEE. https://doi.org/10.48550/arXiv.1904.11483
26. Krasowski H, Thumm J, Müller M, Wang X, Althoff M. Provably safe reinforcement learning: A theoretical and experimental comparison. arXiv preprint arXiv:220506750. 2022. https://doi.org/10.48550/arXiv.2205.06750
27. Yoshida H, Shinohara S, Nagai M. Lane change steering manoeuvre using model predictive control theory. Vehicle System Dynamics. 2008;46(S1):669-81. https://doi.org/10.1080/00423110802033072
28. Singh ASP, Nishihara O. Minimum resultant vehicle force optimal state feedback control for obstacle avoidance. IEEE Transactions on Control Systems Technology. 2019;28(5):1846-61. 10.1109/TCST.2019.2926946
29. Niu J, Hu Y, Jin B, Han Y, Li X, editors. Two-stage safe reinforcement learning for high-speed autonomous racing. 2020 IEEE international conference on Systems, Man, and Cybernetics (SMC); 2020: IEEE. https://doi.org/10.1109/SMC42975.2020.9283053
30. Ohta Y, Maeda H, Kodama S. Reachability, observability, and realizability of continuous-time positive systems. SIAM Journal on Control and Optimization. 1984;22(2):171-80. https://doi.org/10.1137/0322013
31. Kousik S, Vaskov S, Bu F, Johnson-Roberson M, Vasudevan R. Bridging the gap between safety and real-time performance in receding-horizon trajectory design for mobile robots. The International Journal of Robotics Research. 2020;39(12):1419-69. https://doi.org/10.48550/arXiv.1809.06746
32. Althoff M. Reachability analysis and its application to the safety assessment of autonomous cars: Technische Universität München; 2010. https://mediatum.ub.tum.de/doc/1287517/document.pdf.
33. Ahmadian N, Khosravi A, Sarhadi P. Integrated model reference adaptive control to coordinate active front steering and direct yaw moment control. ISA Transactions. 2020;106:85-96. https://doi.org/10.1016/j.isatra.2020.06.020
34. Rasekhipour Y, Khajepour A, Chen S-K, Litkouhi B. A potential field-based model predictive path-planning controller for autonomous road vehicles. IEEE Transactions on Intelligent Transportation Systems. 2016;18(5):1255-67. https://doi.org/10.1109/TITS.2016.2604240
35. Rajamani R. Vehicle dynamics and control: Springer Science & Business Media; 2011. https://doi.org/10.1007/978-1-4614-1433-9
36. Raeesi H, Hosseinpour A, editors. Routing of vehicles by intelligent algorithms in the matter of transporting goods. 2022 IEEE International Conference on Electrical Engineering, Big Data and Algorithms (EEBDA); 2022 25-27 Feb. 2022. 10.1109/EEBDA53927.2022.9744800
37. Al-Dahhan MRH, Schmidt KW. Voronoi boundary visibility for efficient path planning. IEEE Access. 2020;8:134764-81. 10.1109/ACCESS.2020.3010819
38. Gismondi F, Possieri C, Tornambe A. A solution to the path planning problem via algebraic geometry and reinforcement learning. Journal of the Franklin Institute. 2022;359(2):1732-54. https://doi.org/10.1016/j.jfranklin.2021.12.003

39. Blažič S, Klančar G. Effective Parametrization of Low Order Bézier Motion Primitives for Continuous-Curvature Path-Planning Applications. Electronics. 2022;11(11):1709. https://doi.org/10.3390/electronics11111709
40. Eshtehardian S, Khodaygan S. A continuous RRT*-based path planning method for non-holonomic mobile robots using B-spline curves. Journal of Ambient Intelligence and Humanized Computing. 2022:1-10. https://doi.org/10.1007/s12652-021-03625-8
41. Mueller MW, Hehn M, Andrea RD. A Computationally Efficient Motion Primitive for Quadrocopter Trajectory Generation. IEEE Transactions on Robotics. 2015;31(6):1294-310. https://doi.org/10.1109/TRO.2015.2479878
42. Jond H, Platos J, Sadreddini Z. Autonomous vehicle convoy formation control with size/shape switching for automated highways. International Journal of Engineering, Transactions B: Applications. 2020. doi: 10.5829/ije.2020.33.11b.07
43. Kuwata Y, Teo J, Karaman S, Fiore G, Frazzoli E, How J, editors. Motion planning in complex environments using closed-loop prediction. AIAA Guidance, Navigation and Control Conference and Exhibit; 2008. http://dx.doi.org/10.2514/6.2008-7166
44. Fogel E, Halperin D, Wein R, Fogel E, Halperin D, Wein R. Minkowksi sums and offset polygons. CGAL Arrangements and Their Applications: A Step-by-Step Guide. 2012:209-40. https://doi.org/10.1007/978-3-642-17283-0_9
45. Raeesi H, Khosravi A, Sarhadi P. Collision avoidance for autonomous vehicles using reachability-based trajectory planning in highway driving. Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering. 2024. https://doi.org/10.1177/09544070231222053
46. Althoff M, Grebenyuk D, Kochdumper N, editors. Implementation of Taylor models in CORA 2018. Proc of the 5th International Workshop on Applied Verification for Continuous and Hybrid Systems; 2018. https://doi.org/10.29007/zzc7
47. Kousik S. Reachability-based trajectory design 2020. https://hdl.handle.net/2027.42/162884
48. Alshiekh M, Bloem R, Ehlers R, Könighofer B, Niekum S, Topcu U, editors. Safe reinforcement learning via shielding. Proceedings of the AAAI conference on artificial intelligence; 2018. https://doi.org/10.48550/arXiv.1708.08611
49. Hunt N, Fulton N, Magliacane S, Hoang TN, Das S, Solar-Lezama A, editors. Verifiably safe exploration for end-to-end reinforcement learning. Proceedings of the 24th International Conference on Hybrid Systems: Computation and Control; 2021. https://doi.org/10.48550/arXiv.2007.01223
50. Saunders W, Sastry G, Stuhlmueller A, Evans O. Trial without error: Towards safe reinforcement learning via human intervention. arXiv preprint arXiv:170705173. 2017. https://doi.org/10.48550/arXiv.1707.05173
51. Guibas LJ, Nguyen AT, Zhang L, editors. Zonotopes as bounding volumes. SODA; 2003. http://dx.doi.org/10.1145/644108.644241
52. Gaßmann V, Althoff M, editors. Implementation of Ellipsoidal Operations in CORA 2022. Proceedings of 9th International Workshop on Applied; 2022. doi:10.29007/nl86
53. Saraoglu M, Morozov A, Janschek K. Mobatsim: Model-based autonomous traffic simulation framework for fault-error-failure chain analysis. IFAC-PapersOnLine. 2019;52(8):239-44. https://doi.org/10.1016/j.ifacol.2019.08.077
54. Raffin A, Hill A, Gleave A, Kanervisto A, Ernestus M, Dormann N. Stable-baselines3: Reliable reinforcement learning implementations. The Journal of Machine Learning Research. 2021;22(1):12348-55. https://github.com/DLR-RM/stable-baselines3
55. Wagener NC, Boots B, Cheng C-A, editors. Safe reinforcement learning using advantage-based intervention. International Conference on Machine Learning; 2021: PMLR. https://doi.org/10.48550/arXiv.2106.09110

Persian Abstract

چکیده

در سال‌های گذشته تحقیقات در حوزه خودروهای خودران از اهمیت ویژه‌ای برخوردار شده است. این امکان وجود دارد که خودروهای خودران در صورت پیاده سازی، به افزایش کیفیت زندگی روزمره کمک زیادی کنند. برای رسیدن به هدف بیان شده، یک مدل راننده ایمن که وسایل نقلیه خودران را کنترل می کند، مورد نیاز است. یادگیری تقویتی یکی از روش های مناسب برای ایجاد این مدل ها می باشد. در این شرایط عوامل الگوریتم یادگیری تقویتی، معمولاً اقدامات تصادفی را در حین آموزش انجام می دهند که خطر ایمنی در هنگام رانندگی خودروی خودران را به همراه دارد. به منظور رفع این مشکل، در این مقاله یک لایه محافظ پیشنهاد شده است که با پیش بینی حالت آینده پس از انجام یک عمل و تعیین ایمن بودن حالت، این لایه محافظ ایمنی عمل انتخابی را تعیین می کند. برای این منظور باید زونوتوپ های قابل دسترس تهیه شود تا در هر مرحله برنامه ریزی، مجموعه قابل دسترس وسیله نقلیه با هیچ مانعی تلاقی نکند. بنابراین ما یک رویکرد یادگیری تقویتی ایمن با زونوتوپ‌های قابل دسترس مبتنی بر محافظ را پیشنهاد می‌کنیم. شبکه مورد نظر حول DDPG تاخیری دولایه (TD3) ساخته و با آن مقایسه شده است. در حین آموزش و اجرا، سیستم های با لایه محافظ، بدون برخورد هستند و کارایی آنها مشابه یا حتی بهتر از شبکه یادگیری تقویتی بدون لایه محافظ است. نشان داده شده است که یک رویکرد یادگیری مبتنی بر محافظ در قادر ساختن عامل به یادگیری عدم پیشنهاد اقدامات ناامن، موثر است. نتایج شبیه‌سازی‌شده نشان می‌دهد که یک وسیله نقلیه خودران با مجموعه ناایمن در مجاورت فضایی که بیشترین پاداش را ارائه می‌کند، با روش پیشنهادی، در مقایسه با روش‌های دیگری که در حال حاضر برای دستیابی به یادگیری تقویتی ایمن در نظر گرفته می‌شوند، بهتر عمل می‌کند.