Safe and Sample-efficient Reinforcement Learning for Clustered Dynamic Environments

Hongyi Chen[1] and Changliu Liu[2]

***Abstract*— This study proposes a safe and sample-efficient reinforcement learning (RL) framework to address two major challenges in developing applicable RL algorithms: satisfying safety constraints and efficiently learning with limited samples. To guarantee safety in real-world complex environments, we use the safe set algorithm (SSA) to monitor and modify the nominal controls, and evaluate SSA+RL in a clustered dynamic environment which is challenging to be solved by existing RL algorithms. However, the SSA+RL framework is usually not sample-efficient especially in reward-sparse environments, which has not been addressed in previous safe RL works. To improve the learning efficiency, we propose three techniques: (1) avoiding behaving overly conservative by adapting the SSA; (2) encouraging safe exploration using random network distillation with safety constraints; (3) improving policy convergence by treating SSA as expert demonstrations and directly learn from that. The experimental results show that our framework can guarantee safety with high probability during training and solve the task with substantially fewer episodes.**

I. INTRODUCTION

Recently, reinforcement learning (RL) shows promising results in a series of artificial domains; but it's still challenging to develop applicable RL algorithms for a real system due to nine challenges discussed in [1]. Our paper focuses on the two challenges: satisfying safety constraints and learning from limited samples.

Ideally, 0-safety violation should be guaranteed during both training and execution as failures are expensive (damage the real robot systems) and dangerous (hurt humans in the environment). Typical safe RL methods for soft safety constraints include risk-sensitive safe RL, lagrangian methods and constrained policy optimization [2, 3], but no guarantees could be derived. Recently, barrier function method is used to provide hard safety guarantee for RL and achieve 0-safety violation in training [4]. However, this method is only tested in stationary environments like inverted pendulum and car following, with simple safety constraints like enforcing the pendulum angle to be within $[-1, 1]$ radians and keeping 2 meters safe distance between cars.

People evaluate the sample efficiency by measuring the amount of data necessary to achieve a certain performance threshold [1]. But collecting sample data in real world is time-consuming and expensive, and RL agents may converge to local optima when the reward is sparse and never reach the performance threshold. In a word, the sample efficiency challenge makes it hard to deploy RL algorithms quickly in real world systems. Exploration methods, like adding action noise, adding parameter space noise (PSN) [5] and using random network distillation (RND) [6] can help to solve sparse-reward problem but it is risky to explore freely in clustered dynamic environment as the systems would fail or break before learning the optimal controller. Leveraging expert demonstration data can accelerate the agent's learning [7], however people usually get suboptimal demonstration as the expert demonstration is hard to access [8]. Recent model-based deep RL approaches show a lot of promise for improving sample efficiency, however, an imperfect dynamics model can degrade the performance of the learning algorithm and lead to suboptimality [9].

In this work, we aim to design a reinforcement learning framework that can learn safely and efficiently even in *clustered dynamic environments*. We use the safe set algorithm (SSA) [10] to ensure safety. SSA has similar structures as barrier function methods, both of which belong to energy function-based safe control [11]. These methods can safe guard any reinforcement learning algorithm, but they can't help to find optimal policies directly, and sometimes make the system behave overly conservative. Thus, we first adapt the projection direction of SSA to generate more efficient control when possible. Also by combining SSA with normal exploration strategies, we can transform these originally unsafe explorations into safe explorations. Moreover, since getting safe expert demonstration is difficult in the real world, we decide to learn from the safe control generated from SSA online to speedup training. The key contributions of this paper are summarized below:

- We propose the SSA based safe RL training framework and prove this framework can guarantee safety with high probability even in clustered dynamic environments, except the cases that no safe control exists.
- We propose three techniques to improve the learning efficiency: adapting the SSA, exploring under safety constraints and learning from SSA demonstration. The numerical results show that we can solve the task with substantially fewer episodes and interactions.

II. PROBLEM FORMULATION

Environment Dynamics The 2D environment contains multiple dynamic obstacles, every obstacle evolves as $\dot{x}_E = f_E(x_E, u_E)$, where the function f_E represents double integrator dynamics, state x_E involves the position and velocity

[1]Hongyi Chen is with the Institute for Robotics and Intelligent Machines, Georgia Institute of Technology, North Ave NW, Atlanta, GA 30332, USA hchen657@gatech.edu

[2]Changliu Liu is with the Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213, USA cliu6@andrew.cmu.edu

of the obstacle and control u_E represents its acceleration, which is uniformly distributed on the predefined interval.

Robot Dynamics Let $x_R \in X \subset R^{n_x}$ be the robot state containing positions and velocities in x, y axis; $u \in U \subset R^{n_u}$ be the control input, which is the accelerations in x, y axis. The robot dynamics are defined as:

$$\dot{x}_R = f(x_R) + g(x_R)u \tag{1}$$

where $f(x_R) = \left[0, \mathbb{I}_2; 0, 0\right] x_R$, $g(x_R) = \left[0; \mathbb{I}_2\right]$. We assume we can access the ground truth form of f and g.

Safety Specification The safety specification requires the robot to stay in a closed subset of state space, called the safe set X_S. The safe set can be presented by a zero-sublevel set of a continuous and piecewise smooth function $\phi_0 : R^{n_x} \to R$, i.e. $X_S = \{x | \phi_0(x) \leq 0\}$. In our problem, ϕ_0 is defined as $d_{min}^2 - d^2$, where d_{min} is the safety distance defined by user and d is the distance from robot to the closest obstacle. For safety metric, we evaluate the percentage of safety violations during training.

Sample Efficiency Specification To evaluate the data efficiency of a particular model, we measure the amount of data necessary to achieve a certain performance threshold:

$$J^{eff} = \min |D_i| \; s.t. R(\texttt{Train}(D_i)) \geq R_{min}. \tag{2}$$

where D_i is the data used for training the RL policy and R_{min} is the desired performance threshold [1].

Problem The core problem of this paper is to achieve safe and sample-efficient RL learning in clustered dynamic uncertain environment. The learned RL policy will map the state (x_R, x_E^c) to control u, where x_E^c means the closest obstacle to the robot. For safety, we need to monitor and modify the nominal control u to keep the system inside the safe set X_S and achieve least safety-violations. For sample efficiency, we need to ensure RL agent wouldn't converge prematurely to a local optimum and learn the optimal controller with fewer training data.

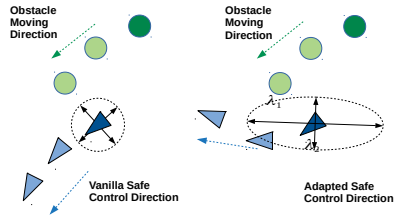
III. Review of the Safe Set Algorithm

The SSA works as a safety monitor [10], which is suitable to safe guard the RL training. The key of SSA is to define a valid safety index ϕ such that 1) there always exists a feasible control input in U that satisfies $\dot{\phi} \leq -\eta\phi$ when $\phi \geq 0$ and 2) any control sequences that satisfy $\dot{\phi} \leq -\eta\phi$ when $\phi \geq 0$ ensures forward invariance of the safe set X_S and finite time convergence to this set. The parameter η is a positive constant that adjusts the convergence rate. Following the safety index design rule [12] for collision avoidance with single obstacle, we define the safety index ϕ as follows:

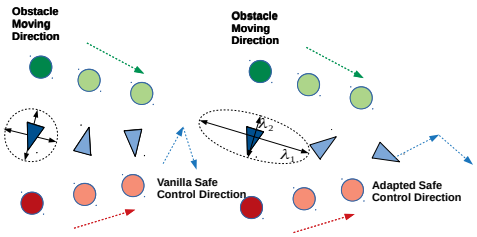
$$\phi = d_{min}^2 - d^2 - k \cdot \dot{d}. \tag{3}$$

where $\dot{d}$ is the relative velocity of robot to obstacle and k is a constant factor. With the valid safety index ϕ, we just need to project the reference control u^r to the set of safe controls that satisfy $\dot{\phi} \leq -\eta\,\phi$ when $\phi \geq 0$, and $\dot{\phi}$ is expressed as

$$\dot{\phi} = \frac{\partial \phi}{\partial x} f + \frac{\partial \phi}{\partial x} g \; u = L_f\phi + L_g\phi \; u. \tag{4}$$



(a) The robot bypasses the obstacle with adapted SSA but is pushed back by the obstacle with vanilla SSA.



(b) The robot escapes from two obstacles with adapted SSA but oscillates between them with vanilla SSA.

Fig. 1: Comparison between vanilla SSA and adapted SSA. The darker color presents the current positions of the robot and the obstacles. The lighter color presents the future positions of the robot and the obstacles.

We compute ϕ_i for every obstacle and add safety constraint whenever ϕ_i is positive. Also we have velocity and acceleration limits. With all these constraints, SSA will solve the following optimization problem through quadratic programming (QP):

$$\begin{aligned} \min_{u \in U} ||u - u^r||^2 &= \min_{u \in U} u^{\mathrm{T}} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} u - 2u^{\mathrm{T}} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} u^r \\ s.t. L_f\phi_i + L_g\phi_i \; u &\leq -\eta\,\phi_i, i = 1, 2...m. \end{aligned} \tag{5}$$

However, in clustered dynamic environment, there are situations that don't even exist safe control to guarantee safety as we will discuss later (note ϕ in (3) only guarantees safety with single dynamic obstacle not multiple dynamic obstacles). Besides, low sample efficiency is a problem in vanilla SSA: the agent may require long training period or even fail to learn optimal controller when the task is complex or the environment is reward-sparse. To make it work, we need to improve the sample efficiency with the following three strategies.

IV. Methodology

A. Adapting the Safe Set Algorithm

Vanilla SSA would output safe control that drives the system to the currently safest direction, which may not be an efficient direction in the long run, see fig. 1a. Besides, in multi-obstacle environment, it's not safe to directly add constraints for every dangerous obstacle whose ϕ is positive. In detail, since vanilla SSA only considers these dangerous obstacles, it may push the robot to the direction that is safe now but risky in the next step if there are unconsidered

obstacles (ϕ values are negative) in that direction, see fig. 1b. Thus we decide to consider the current positions, estimate future positions of all approaching obstacles, and modify the direction of safe control by tuning parameters in the QP problem (6). After adapting the projection direction, we expect to generate control signal that is safe to all approaching obstacles, even these ϕ values are negative, and efficient for the longer time horizon.

$$\min_{u\in U} ||u-u^r||_Q = \min_{u\in U} u^{\mathrm{T}} \begin{bmatrix} \alpha & \sigma \\ \sigma & \beta \end{bmatrix} u - 2u^{\mathrm{T}} \begin{bmatrix} \alpha & \sigma \\ \sigma & \beta \end{bmatrix} u^r. \quad (6)$$

In detail, we define the approaching obstacles as those whose distances to the robot are smaller than a threshold value. With the current position of the robot (x_0, y_0) and the current positions of approaching obstacles $(x_i, y_i), i = 1, 2, ..., n$, we first predict the next k steps positions of each obstacle $(x_i^j, y_i^j), j = 1, 2, ..., k$ using constant velocity model. Then we solve the line $l_\theta : -\sin(\theta)x + \cos(\theta)y = 0$ that maximizes the distance to all approaching obstacles

$$\max_\theta J(\theta), \quad J(\theta) := \sum_{j=1}^{k} \sum_{i=1}^{n} d_i^j. \quad (7)$$

where d_i^j is the distance of shifted obstacle $(x_i^j - x_0, y_i^j - y_0)$ to the line l_θ. In this setting, we regard the robot system (x_0, y_0) as the origin and calculate $\theta_{\max}$ that has largest overall distance. With $\theta_{\max}$, we can get eigenvector $\boldsymbol{x}_1 = (-\sin(\theta_{\max}), \cos(\theta_{\max}))$, which is the safest direction for all approaching obstacles, and $J(\theta_{\max})$ is its corresponding eigenvalue λ_1. The larger λ_1 is, the more we want to project the safe control to $\boldsymbol{x}_1$ direction. The second eigenvector $\boldsymbol{x}_2$ is perpendicular to $\boldsymbol{x}_1$ and has smallest overall distance λ_2. Then, we can build the QP parameter matrix, which is represented as the ellipse in fig. 1a and fig. 1b, as follows:

$$[\boldsymbol{x}_1, \boldsymbol{x}_2] \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} [\boldsymbol{x}_1, \boldsymbol{x}_2]^{-1} = \begin{bmatrix} \alpha & \sigma \\ \sigma & \beta \end{bmatrix} =: Q. \quad (8)$$

B. Exploration under Safety Constraints

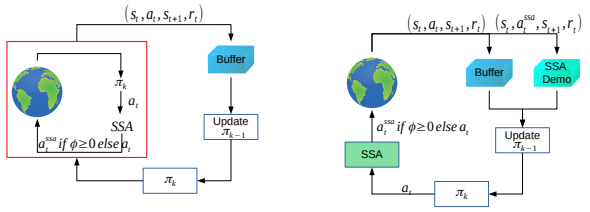
In real world reward-sparse and clustered dynamic environments, it's challenging to find a sequence of actions that can lead to positive reward and generalize to related situations, thus RL agents need long training time. Traditional exploration techniques used to address this problem are not suitable for safety-critical tasks, as they may explore unsafe controls. In our framework, with the help of SSA, we add safety constraints to the following two exploration strategies to improve the suboptimal policy safely.

Parameter Space Noise (PSN) [5] At the start of each episode, we create a copy of RL policy and add noise directly to the policy's parameters, which can lead to consistent exploration and a richer set of behaviors. Suppose we parameterize the policy π_θ as a neural network with weights θ. Then the exploration policy is $\pi_{\widetilde{\theta}}$, where

$$\widetilde{\theta} = \theta + N(0, \sigma^2 I) \quad (9)$$

Random Network Distillation (RND) [6] This exploration strategy will modify the reward function to encourage the agent to visit novel states. In detail, we create two neural networks that take the state $s = (x_R, x_E^c)$ as input and train one of the networks to predict the output of the other. The prediction error of two neural networks is defined as novelty and will be added to reward:

$$\widetilde{r}(s, a) = r(s, a) + ||f_{\theta_1}(s) - f_{\theta_2}(s)||_2^2 \quad (10)$$



(a) SSA+RL Default Learning (b) SSA+RL Safe Learning

Fig. 2: Block diagrams of the default learning and proposed safe learning. In default learning, the environment (the red box) contains the world and SSA module, while in safe learning, SSA is separated from the environment.

C. Learning from SSA Safe Demonstration

Another technique people use to improve sample efficiency is learning from demonstration (LfD) instead of learning from scratch. Different from the traditional LfD or safe controller guided training in [4], we don't need to prepare demonstration data and pre-train the RL agent or approximate all prior safe controllers. Instead, SSA would generate safe controls during the interactions with the world and these safe controls are regarded as expert demonstration. To be more specific, in default SSA+RL framework fig. 2a, SSA is part of the environment which means the RL agent's control signal will be modified when $\phi \geq 0$ and the agent wouldn't realize that. While in safe learning, we separate the SSA from the environment and make it an independent module, see fig. 2b. In this way, the agent could know the world is taking a_t or a_t^{ssa}, then store the self-generated data and demonstration data into two buffers. When updating the policy, we simply use a fixed ratio between self-generated data and SSA demonstrations to mix the training samples. We will explore the optimal ratio in future work.

V. Experiments

A. Environment of Autonomous Driving Task

We evaluate our proposed framework in a clustered dynamic environment with sparse reward. Our goal is to move the vehicle, starting from the bottom, to the green area on the top while avoiding 50 moving obstacles in between, which is challenging to be solved by the state-of-the art RL algorithms alone. A success case is shown in fig. 3a. In this task, success means the vehicle arrives at goal within 1000 steps and receives 2000 reward while failure means the vehicle doesn't reach the goal within the time limit and gets 0 reward. Collision means the vehicle collides with obstacles and gets -500 penalty. We assume the vehicle can sense the correct positions and velocities of obstacles, but

doesn't know their accelerations, which introduce uncertainty to the environment. Also the state of every obstacle will be randomly initialized at each episode. This environment tries to simulate the real world scenarios like parking lots and busy streets that have multiple dynamic objects moving around.

B. Experiment Design

Hypothesis: we evaluate the proposed framework by verifying the following four hypotheses:

- H1: The SSA+RL framework can greatly reduce safety violation comparing to the TD3 baseline model in clustered dynamic environment.
- H2: The adapted SSA can achieve better efficiency and higher task success rate than vanilla SSA.
- H3: Traditional exploration strategies are not safe and hence not data efficient, but exploration under safety constraints could improve the sample efficiency.
- H4: Direct learning from the safe controls demonstrated by SSA can best speedup training and maintain safety compared to pure reward-driven approaches.

C. Baseline Model and Evaluation Method

We adopt the Twin Delayed Deep Deterministic Policy Gradients (TD3) [13] as our baseline RL model. TD3 is built upon the DDPG algorithm and tries to address the overestimation bias problem with DDPG by using two critics Q_{θ_1}, Q_{θ_2} and two corresponding targets $Q_{\theta'_1}$, $Q_{\theta'_2}$. For both critics, the target update is computed using the minimum of two target networks, which successfully reduces the overestimation bias.

For experiments that evaluate safety, we train the RL policy combining with different proposed techniques for 50 episodes as we notice 50 episodes are usually long enough for the SSA-based models to converge. For experiments that evaluate sample efficiency, we train our models until their performances reach a threshold reward R_{min} or the training go over the maximum number of episodes. The required R_{min} is to achieve at least 1900 on average for the past 20 episodes. The maximum training episodes and environment interaction steps are 1000 and 10^6 respectively. For models that fail to meet R_{min} within 1000 episodes, we set its result as 1000 episodes and 10^6 interactions. To guarantee the robustness of our experiments, we repeat each training with different seeds for 10 times and calculate the average performance. The code is open sourced `here`.

D. Results

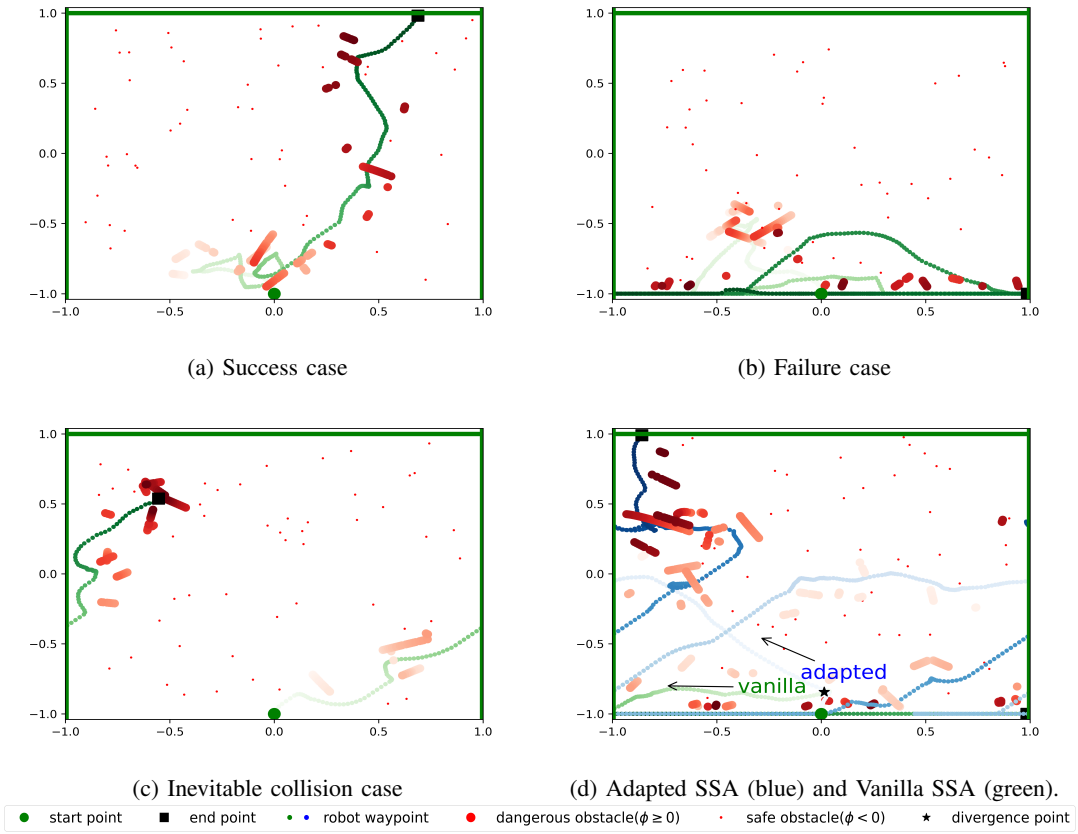
H1: Through our experiments, TD3 model gets 31.7% collision rate and 66.3% failure rate on average, see table I. During training, the policy gradually converges to a local optimum after repeated collisions, which is staying at the bottom where obstacles can't get to. As shown in fig. 3b, even though the vehicle tries to navigate towards the goal area, it is pushed back by obstacles and spends the remaining time at the bottom to avoid collision and penalty. That's why the failure rate is even higher than the collision rate.

On the other hand, SSA helps to reduce the collision rate from 31.7% to 0.8%. In our evaluation environment, we find SSA can't achieve 0-safety violation discussed in other papers using static testing cases like avoiding fixed obstacles or static hazard areas; in which cases, these always exist a safe control to meet safety constraints [4][12]. But in our environment, the obstacles are not only dynamic but may move in an unpredictable way as well. What's worse, the vehicle can be surrounded by multiple obstacles and there does not exist a safe control to meet all safety constraints. In this case, the collision is inevitable as shown in fig. 3c. There were three obstacles driving from three different directions toward the vehicle, and SSA cannot find a safe control and collision happens. Besides, we notice that with SSA, the success rate increased from 2% to 50.2%. The main reason is that SSA prevents the RL agent from collisions and increases the possibility of reaching the goal. Such results qualitatively show that SSA can greatly improve safety of RL training in complex environment. But SSA+RL may still stuck in local optimums as its failure rate is as high as 49%.

TABLE I: Safety comparison between all models.

	Model	Success	Failure	Collision	Reward
Baseline Models	RL	2%	66.3%	31.7%	-118.6
	PSN+RL	3.6%	76.8%	19.6%	-26
	RND+RL	5.2%	49.5%	47.8%	-133.8
Proposed Models	SSA+RL	50.2%	49%	**0.8%**	1000
	Adapted SSA+RL	68.6%	30%	1.4%	1365
	PSN+SSA+RL	40.4%	58.3%	1.3%	800
	RND+SSA+RL	71.4%	27.2%	1.4%	1421
	LfD+SSA+RL	**89.8%**	**9.4%**	**0.8%**	**1792**
	Penalty+SSA+RL	43.8%	55.2%	1.0%	871

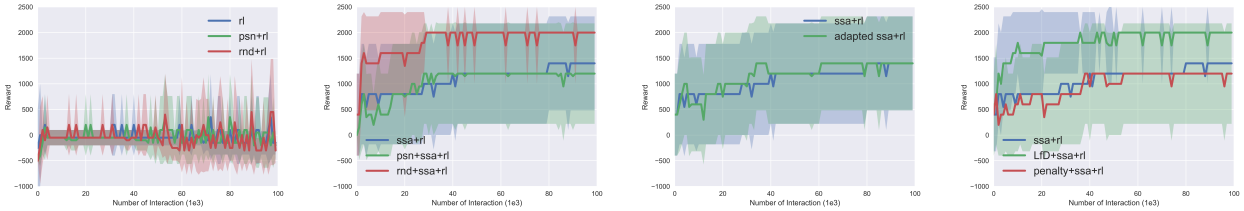
H2: The improvement of SSA adaptation is less significant compared to other techniques we used due to two main reasons. Firstly, SSA adaption only works when SSA is triggered, which wouldn't happen at every step. Secondly, in many cases, vanilla SSA and adapted SSA will generate very close safe controls as both of them need to satisfy the same hard safety constraint when solving the optimization problem. As shown in table III, the averaged interaction number of adapted SSA+RL model is 2.3×10^5, which is very close to the interaction number 2.4×10^5 of SSA+RL model, because the adapted SSA+RL model may also get stuck in the local optimum as vanilla SSA+RL does. Nevertheless, we find the failure rate drops from 49% to 30% and the success rate increases from 50.2% to 68.6% in table I. This improvement is due to that adapted SSA generates fewer detours when meeting dangerous obstacles, and the projection guidance leads the vehicle to a more efficient direction. The effects of adapted safe controls will accumulate and improve the overall success rate and reward. In fig. 3d, adapted SSA and vanilla SSA are tested in the same environment with the same random seed. At the beginning, their paths are identical, but diverge gradually. The black star is the most critical divergence point between them. When meeting dangerous obstacle at that star point, the adapted SSA outputs control in upwards direction while the vanilla SSA outputs control



(a) Success case (b) Failure case

(c) Inevitable collision case (d) Adapted SSA (blue) and Vanilla SSA (green).

Fig. 3: Trajectory plots of four cases. For both vehicle and obstacles, the darker color means more recent position while the lighter color means older position. The vertical green lines are boundaries that will wrap around. The top green line represents the goal area.



(a) Baseline Models (b) Exploration with SSA (c) Adapted SSA (d) Learning from SSA

Fig. 4: Average performance of baseline models and our proposed models over 10^6 steps interactions.

in downwards direction. At last, the adapted SSA helps the vehicle reach the goal while the vanilla SSA pushes the vehicle back to the bottom.

H3: This hypothesis can be proved via results in table I. The exploration strategies PSN and RND only improve the success rate slightly from 2% to 3.6% and 5.2% respectively. But the failure rate of PSN-enabled model increases from 66.3% to 76.8% and the collision rate of RND-enabled model increases from 31.7% to 47.8%. That is mainly because these exploration strategies have large portion of harmful exploration, i.e., unsafe controls that don't meet the safety constraint $\dot{\phi} \leq -\eta\,\phi$. To further validate this, we measure the percentages of safe control and unsafe control when meeting dangerous obstacles ($\phi \geq 0$) in table II. Traditional explorations have high probability ($\geq 68\%$) to take unsafe controls that may cause collisions and this probability increases when the number of obstacles in the environment increases.

Combining with SSA, the percentage of unsafe control drops to 2% even in most complex situation. The RND+SSA+RL model can reduce the failure rate from 49% to 27.2% and increase the success rate from 50.2% to 71.4%, which mitigates the problem of being overly conservative in clustered environment. Moreover, in table III, this model uses substantially fewer episodes and 38$\times$ fewer environment interactions to meet R_{min} compared to the SSA+RL model. By visiting new states, the agent has more chance to avoid getting stuck in the local optimum and converge to optimal policy faster. However, PSN+SSA+RL gets worse perfor-

mance than SSA+RL, which may because PSN is unable to sufficiently explore in our challenging environments.

TABLE II: Percentages of safe and unsafe controls when meeting dangerous obstacles ($\phi \geq 0$) in different environment complexities.

Models	Env Complexity	Unsafe Action ($\dot{\phi} > -\eta\,\phi$)	Safe Action ($\dot{\phi} \leq -\eta\,\phi$)
Exploration+RL	30 Obstacles	68.8%	31.2%
	50 Obstacles	76.2%	23.8%
	70 Obstacles	75.6%	24.4%
Exploration+ SSA+RL	30 Obstacles	1.1%	98.9%
	50 Obstacles	1.0%	99.0%
	70 Obstacles	2.0%	98.0%

H4: For learning from demonstration version of SSA+RL, even we don't have expert demonstration in advance, the SSA generated online safe demonstration (s_t, a_t^{SSA}) is good enough to bootstrap the RL training. In table III, it uses only 26 episodes and 8464 steps to meet R_{min}, using 28$\times$ fewer steps compared to SSA+RL model. From fig. 4d, we find safe learning can help the model achieve faster convergence and greater training stability. This model learns the optimal controller in all experiments, while other models may still get 0 reward after long training episodes and have big deviations. Moreover, unlike other techniques that will increase the collision rate slightly, LfD+SSA+RL achieves highest success rate 89.8% and lowest collision rate 0.8% in table I, which proves that learning from SSA demonstration can best maintain safety. This is due to two reasons: Firstly, there is a mismatch between the control that RL agent generates and the real control that environment takes due to control modification in the default SSA+RL framework. This will introduce errors to the training data and lower the training efficiency. Secondly, the RL policy could better learn how to take safe control with SSA demonstration, instead of learning from scratch or the reward penalty. To validate the second point, we give the agent a negative reward penalty $-||a_t^{ssa} - a_t||_2^2$ to negatively reinforce unsafe control following the idea in [14]. The results show that penalty+SSA+RL model has lower success rate 43.8%, slightly higher collision rate 1.0% and takes more interactions steps compared to SSA+RL model in table I and table III. This is because learning safe control from the reward penalty is too hard for the agent, which reduces the learning efficiency.

TABLE III: Efficiency comparison between all models.

	Model	Episodes	Interaction
Baseline Models	RL	1000	10^6
	PSN+RL	1000	10^6
	RND+RL	1000	10^6
Proposed Models	SSA+RL	254	2.4×10^5
	Adapted SSA+RL	245	2.3×10^5
	PSN+SSA+RL	251	2.4×10^5
	RND+SSA+RL	**23**	**6183**
	LfD+SSA+RL	26	8464
	Penalty+SSA+RL	331	3.2×10^5

VI. Conclusion

In this work, we propose to use SSA to improve safety during RL policy training, and introduce three strategies, including SSA adaption, exploration under safety constraint and learning from SSA demonstrations, to improve the learning efficiency. We further validate the proposed framework in a clustered dynamic environment. The results show that SSA can greatly reduce the safety-violation except for the situations that no safe control exists. Combining with the three strategies, the agent can solve the task with substantially fewer episodes and interactions.

This framework can be applied to a wide range of safety-critical learning tasks, like learning locomotion skills and keeping balance for bipedal robots, learning autonomous driving in crowded streets. In future work, we will consider two aspects: firstly, we will introduce robust SSA to guarantee safety under uncertainties. Secondly, we will adopt SSA in complicated physical robots, like bipedal robots whose state space is high dimension and dynamics are complex.

References

[1] C. Paduraru et al. "Challenges of Real-World Reinforcement Learning:Definitions, Benchmarks Analysis". In: *Machine Learning Journal* (2021).

[2] P. Geibel and F. Wysotzki. "Risk-Sensitive Reinforcement Learning Applied to Control under Constraints". In: *Journal of Artificial Intelligence Research* (2005), 81–108.

[3] J. Achiam et al. "Constrained Policy Optimization". In: *Proc. Int. Conf. Machine Learning*. 2017, 22–31.

[4] R. Cheng et al. "End-to-End Safe Reinforcement Learning through Barrier Functions for Safety-Critical Continuous Control Tasks". In: *Proc. AAAI Conf. Artificial Intelligence*. 2019, pp. 3387–3395.

[5] M. Plappert et al. "Parameter Space Noise for Exploration". In: *Proc. Int. Conf. Learning Representations*. 2018.

[6] Y. Burda et al. "Exploration by random network distillation". In: *Proc. Int. Conf. Learning Representations*. 2019.

[7] T. Hester et al. "Learning from Demonstrations for Real World Reinforcement Learning". In: *arXiv preprint arXiv:1704.03732* (2017).

[8] A. Mandlekar et al. "IRIS: Implicit Reinforcement without Interaction at Scale for Learning Control from Offline Robot Manipulation Data". In: *Proc. IEEE Conf. Robotics and Automation*. 2020, pp. 4414–4420.

[9] J. Buckman et al. "Sample-Efficient Reinforcement Learning with Stochastic Ensemble Value Expansion". In: *Proc. Int. Conf. on Neural Information Processing Systems*. 2018, 8234–8244.

[10] C. Liu and M. Tomizuka. "Control in a safe set: Addressing safety in human-robot interactions". In: *Dynamic Systems and Control Conference*. 2014, V003T42A003.

[11] T. Wei and C. Liu. "Safe Control Algorithms Using Energy Functions: A Uni ed Framework, Benchmark, and New Directions". In: *Proc. IEEE Conf. Decision and Control*. 2019, pp. 238–243.

[12] Anonymous. "Model-free Safe Control for Zero-Violation Reinforcement Learning". In: *Submitted to 5th Annual Conference on Robot Learning*. 2021.

[13] S. Fujimoto, H. van Hoof, and D. Meger. "Addressing Function Approximation Error in Actor-Critic Methods". In: *Proc. Int. Conf. Machine Learning*. 2018, pp. 1582–1591.

[14] W. Saunders et al. "Trial without Error: Towards Safe Reinforcement Learning via Human Intervention". In: *Proc. Int. Conf. Autonomous Agents and MultiAgent Systems*. 2018.