A Dynamic Safety Shield for Safe and Efficient Reinforcement Learning of Navigation Tasks

Murad Dawood DAWOOD@CS.UNI-BONN.DE

Ahmed Shokry SHOKRY@CS.UNI-BONN.DE

Maren Bennewitz MAREN@CS.UNI-BONN.DE
Humanoid Robots Lab, University of Bonn, Germany

Editors: N. Ozay, L. Balzano, D. Panagou, A. Abate

Abstract

Reinforcement learning (RL) has been successfully applied to a variety of robotics applications, where it outperforms classical methods. However, the safety aspect of RL and the transfer to the real world remain an open challenge. A prominent field for tackling this challenge and ensuring the safety of the agents during training and execution is safe reinforcement learning. Safe RL can be achieved through constrained RL and safe exploration approaches. The former learns the safety constraints over the course of training to achieve a safe behavior by the end of training, at the cost of high number of collisions at earlier stages of the training. The latter offers robust safety by enforcing the safety constraints as hard constraints, which prevents collisions but hinders the exploration of the RL agent, resulting in lower rewards and poor performance. To overcome those drawbacks, we propose a novel safety shield, that combines the robustness of the optimization-based controllers with the long prediction capabilities of the RL agents, allowing the RL agent to adaptively tune the parameters of the controller. Our approach is able to improve the exploration of the RL agents for navigation tasks, while minimizing the number of collisions. Experiments in simulation show that our approach outperforms state-of-the-art baselines in the reached goals-to-collisions ratio in different challenging environments. The goals-to-collisions ratio metrics emphasizes the importance of minimizing the number of collisions, while learning to accomplish the task. Our approach achieves a higher number of reached goals compared to the classic safety shields and fewer collisions compared to constrained RL approaches. Finally, we demonstrate the performance of the proposed method in a real-world experiment.

Keywords: Safe Reinforcement Learning, Navigation, Model Predictive Control

1. Introduction

In the last decade, reinforcement learning (RL) has been shown to outperform classical methods in navigation tasks [Xu et al. (2023); He et al. (2024)]. This is mainly due to the fact that RL approaches are able to directly map sensory information into actions, enabling them to learn in complex scenarios that would otherwise necessitate lots of engineering efforts. Nevertheless, the safety aspect of RL and the transfer of the trained policies to the real-world remain challenging.

Safe RL has been proposed to ensure the safety of the RL agents during training and execution and two main directions have emerged in the field. First, constrained reinforcement learning: this approach formulates the RL as a constrained Markov decision problem to minimize safety constraints violations during training [Achiam et al. (2017); Tessler et al. (2018); Stooke et al. (2020);

Agha et al. (2024)]. This method trains an additional network to estimate the cost of constraint violations. The policy is then trained to maximize rewards while minimizing those costs. Although these methods often achieve safe behavior by the end of training, they tend to violate constraints frequently during early learning. Additionally, the end-to-end nature of these policies makes their transfer to the real-world even more challenging due to the sim-to-real gap.

The second approach to achieve safe RL is safe exploration. In addition to the RL agent responsible for achieving the task (task agent), a safe policy, also referred to as a safety shield, is introduced to ensure the safety of the task agent during training and execution [Wabersich et al. (2021); Dalal et al. (2018); Zhang et al. (2023)]. The safety shield utilizes knowledge about the dynamics of the robot, without having knowledge about the dynamics of the environment, to override unsafe actions from the task agent during training and execution. Although, this approach is more reliable and suffers from less constraints violations, it tends to overly restrict the exploration process, which consequently leads to less rewards achieved by the RL agents. This is caused by the formulation of the safety shields, which focus mainly on collisions avoidance and perform with limited prediction horizons.

To address these shortcomings, we propose integrating the long-horizon predictive capabilities of reinforcement learning with the robustness of optimization-based controllers, see Fig. 1. Specifically, we combine a model predictive control (MPC)-based safety shield with an RL agent to provide enhanced guidance to the task agent. This agent is referred to as the supervisor agent and is responsible for dynamically adjusting the weights of the constraints based on the current observations. Additionally, to ensure that the safety shield does not overly constrain the task agent's exploration, the supervisor agent also adjusts the weights to align the shield's actions with those of the task agent. The supervisor agent is responsible for avoiding collisions while matching the actions of the task agent, without having knowledge about the main task of the task agent. A clear advantage of not informing the supervisor agent about the main task is to minimize the exploration needed for the supervisor agent to converge. This results in a small number of constraints violations, while not affecting the exploration of the task agent, as we show in the experiments.

In this work, we focus specifically on safety in navigation tasks, where the task agent controls the robot's linear and angular velocities. Thus, the considered constraints are related to obstacle avoidance and constraints violations are collisions done by the learning robot.

In summary, the main contributions of our work are: **(i)** A novel safety shield for navigation tasks that combines the robustness of the MPC shields with the long-horizon capabilities of the RL agents, by allowing an additional RL agent to dynamically adjust the weights of the constraints and the weights for matching the task agent's actions. **(ii)** A supervisor RL agent without access to goal-related information to dynamically adjust the MPC shield online. This goal-independent training results in small number of collisions as we show in the ablation study. **(iii)** We show over several simulation environments of increasing difficulties the superiority of our approach over several state-of-the-art baselines w.r.t. the reached goals-to-collisions ratio. Finally, we demonstrate the performance of our approach in real-world scenarios.

2. Related Work

2.1. Constrained Reinforcement Learning

Several studies proposed formulating the RL task as a constrained Markov decision problem, to take the constraints into consideration during training. In [Achiam et al. (2017)] the authors modified

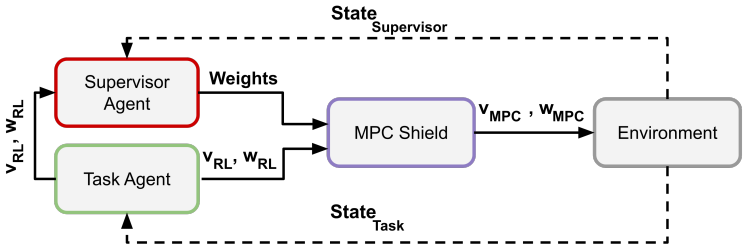


Figure 1: Architecture of our approach. The task agent (green) is responsible for learning the navigation task. The agent receives the $State_{Task}$ from the environment and outputs the linear and angular velocities (v_{RL}, w_{RL}). The supervisor agent (red) receives the $State_{Supervisor}$ from the environment and outputs the *Weights* for aligning the MPC-shield's actions with the task agent's actions, and the weights of the constraints. The MPC shield solves the optimal control problem (Eq. 2) using the all mentioned weights to find the safe actions v_{MPC}, w_{MPC}.

the trust-region policy optimization (TRPO) method to include the constraints on expected costs in the update rule of the algorithm. The study [Ray et al. (2019)] introduced the Lagrangian TRPO and Lagrangian proximal policy optimization methods, and showed that constrained RL methods achieve fewer constraints violations compared to the unconstrained RL at the cost of reduced rewards. However, a common issue in the Lagrangian constrained RL methods was instability during training. In [Stooke et al. (2020)] the authors addressed this instability by introducing PID control over the Lagrangian update and showed improved performance. [Thananjeyan et al. (2021)] suggested pre-training a separate recovery policy along with a critic that learns the constraints violations offline then continues the training online. The pre-training phase requires a replay buffer that demonstrates examples of constraints violations. [Sootla et al. (2022)] proposed using a safety budget, which represents the remaining allowable constrains violation, instead of the cost to reduce constraints violations. [Agha et al. (2024)] combined model-based constrained RL with the recovery policy from [Thananjeyan et al. (2021)] and showed reduction in the constraint violation compared to several model-free constrained RL baselines. These methods learn the constraints during training to decrease the rate of constraints violation over the course of training, which results in high number of constraints violations during the initial training stages.

2.2. Safe Exploration

On the other spectrum of safe RL lie safe exploration approaches, which promote safer exploration by eliminating actions that would lead to unsafe states. [Wabersich and Zeilinger (2021)] proposed using MPC-based safety shield to override unsafe actions from the RL agent, where the safe states were formulated as hard constraints. [Dalal et al. (2018)] pre-trained a neural network to act as the safety shield, and added it to the policy network that is trained from scratch. [Carr et al. (2023)] used a state estimator in addition to the safety shield to improve the agent safety. [Zhang et al. (2019)] utilized predictive safety shields to ensure the safety of the agents in multi-agents scenarios. [Zhang et al. (2023)] used control barrier functions [Ames et al. (2019)] as a shielding mechanism to ensure the safety of the agents. Different from these methods, we take advantage of the RL long-horizon prediction capabilities to tune the safety shield used in training the RL task agent. This leads to a less constrained shield that does not hinder the exploration of the task agent as we show in the experiments. Moreover, softening the hard constraints and tuning their weights online leads to less failures of the MPC solver, where the MPC is unable to find a solution, which commonly happens when considering several constraints as in [Brito et al. (2021); Dawood et al. (2025)].

3. Our Approach

In this work, we propose a novel safety shield that is tuned online using an RL agent, i.e., the supervisor agent, to ensure the safety of the task agent, see Fig.1. The shield along with the supervisor agent do not have access to goal-related information, which is handled by the task agent. Our aim is to develop a safety shield that minimizes the number of collisions, while not hindering the exploration process of the task agent.

First, we introduce the considered navigation task. We then explain the dynamic safety shield and the differences from the classic MPC-safety shield [Wabersich and Zeilinger (2021); Dawood et al. (2025)]. Afterwards, we introduce the supervisor RL agent responsible for adjusting the shield online. Finally, we introduce the task agent used for solving the task.

3.1. Navigation Task

We focus on the navigation task, where the RL task agent should learn how to navigate to goals, as fast as possible, without collisions in unknown environments. We assume that the robot is equipped with a lidar sensor and is given the relative distance and heading to the goal. The lidar sensor provides 100 equidistant beams over a range of 360°, giving obstacle information around the robot. The task agent uses this information to produce the linear and angular velocities for the robot.

3.2. Dynamic Model Predictive Control Safety Shield

The MPC shield requires a mathematical model of the robot to predict the future states based on the current state and the calculated control sequence. An optimization control problem is solved at each time step to find the control sequence that minimizes a predefined cost function. Only the first control action in the control sequence is applied to the robot. It is important to note that the mathematical model represents the kinematics of the robot only and does not include the dynamics of the world model as in model-based RL [Huang et al. (2023); Agha et al. (2024)].

Prediction Model: The discrete-time model of the robot is: $\mathbf{x}_{t+1} = \mathbf{x}_t + \begin{bmatrix} \cos\theta_t & 0 \\ \sin\theta_t & 0 \\ 0 & 1 \end{bmatrix} \mathbf{a}_t \Delta t$, where $\mathbf{x} = [x, y, \theta]^T$ represents the state of the robot, which is the position and heading relative to the starting position. $\mathbf{a} = [v, \omega]^T$ is the action, which is the linear and angular velocities of the robot.

Optimal Control Problem (OCP): To ensure the safety of the robot while minimizing the intervention of the safety filter, the optimization problem at time step t is formulated as:

$$\min_{\substack{\mathbf{x}_{t:t+T|t}, \\ \mathbf{a}_{t:t+T-1|t}}} \|\mathbf{a}_{RL} - \mathbf{a}_{t|t}\|^2_{R_0} + \sum_{k=1}^{T-1} \|\mathbf{a}_{t+k|t}\|^2_R \tag{1a}$$

$$\text{s.t.} \quad \mathbf{x}_{t|t} = \mathbf{x}_t, \tag{1b}$$

$$\mathbf{x}_{t+k+1|t} = f(\mathbf{x}_{t+k|t}, \mathbf{a}_{t+k|t}), \forall k = 0, 1, ..., T-1 \tag{1c}$$

$$\mathbf{a}_{t+k|t} \in \mathbf{A}, k = 0, 1, ..., T-1 \tag{1d}$$

$$\mathbf{x}_{t+k|t} \in \mathbf{X}, \forall k = 0, 1, ..., T-1, \tag{1e}$$

$$dist^{t+k|t}_{obst} > \delta, \forall obst = 1, 2..., M, \forall k = 0, 1, ..., T-1, \tag{1f}$$

where T represents the prediction horizon. The weighted Euclidean norm, represented by $\|\cdot\|_R^2$, is defined as $\|\mathbf{x}\|_R^2 = \mathbf{x}^\mathsf{T} R\mathbf{x}$, where R is a positive definite weighting matrix. The notation $t+k|t$ indicates predictions at time $t+k$, assuming the current time is t. The first term in (1a) measures the deviation between the task agent's proposed action $\mathbf{a}_{RL}$ (Action$_{RL}$) and the initial MPC action $\mathbf{a}_t$ (Action$_{MPC}$). The second term minimizes the magnitude of future control signals $\mathbf{a}_{t+k}$, promoting smoother transitions. Weight matrices R_0, R are used to optimize the performance of the shield and are manually tuned. Constraints (1b–1f) are hard constraints: (1b) ensures the initial state of the model matches the actual state of the robot, (1c) enforces the robot's dynamic model, (1d) and (1e) bounds the actions and states within the predefined limits, where $\mathbf{X}$ and $\mathbf{A}$ denote the allowable sets of states and controls, respectively. (1e) enforces the distance between the robot and the nearest M obstacles to be larger than a safety threshold distance δ. M is a design parameter representing the maximum number of obstacles considered by the MPC. Since the MPC process raw lidar data directly, the lidar data is divided into M equal sectors and only the beam with the smallest distance in each sector is considered as obstacle. In this work, we use $M = 4$.

The main drawbacks of the above formulation are that considering several obstacles as hard constraints in the MPC problem can often lead the solver to fail to find a feasible solution online as in [Brito et al. (2021)], and manually tuning the weight matrix R_0 to achieve a reasonable performance is a time-consuming process. That is why we propose modifying the OCP as follows:

$$\min_{\substack{\mathbf{x}_{t:t+T|t},\\ \mathbf{a}_{t:t+T-1|t}}} \|\mathbf{a}_{RL} - \mathbf{a}_{t|t}\|_{R_0}^2 + \sum_{k=1}^{T-1} \|\mathbf{a}_{t+k|t}\|_R^2 + \sum_{obst=1}^{M} \frac{\omega_{obst}}{dist_{obst}^{t+k|t}} \tag{2a}$$

$$\text{s.t.} \quad \mathbf{x}_{t|t} = \mathbf{x}_t, \tag{2b}$$

$$\mathbf{x}_{t+k+1|t} = f(\mathbf{x}_{t+k|t}, \mathbf{a}_{t+k|t}), \forall k = 0, 1, ..., T-1 \tag{2c}$$

$$\mathbf{a}_{t+k|t} \in \mathbf{A}, k = 0, 1, ..., T-1 \tag{2d}$$

$$\mathbf{x}_{t+k|t} \in \mathbf{X}, \forall k = 0, 1, ..., T-1, \tag{2e}$$

where the green terms in the cost function 2a indicates our modifications. The weight matrix R_0 is now tuned online, while the obstacle avoidance terms scaled by the weights ω_{obst} are now added to the cost function. However, setting these weights equally often leads the robot to being stuck when surrounded by obstacles [Dawood et al. (2025)]. That is why we learn these weights online based on the observations of the supervisor agent which we discuss in the next section. We do not tune the R matrix online, We do not tune the R matrix online, as its purpose is to reduce the control effort over the rest of the prediction horizon.

3.3. Supervisor Reinforcement Learning Agent

To adjust the MPC shield online, we use a soft-actor-critic [Haarnoja et al. (2018)] (SAC) agent, which we call the supervisor agent. The role of the supervisor agent is to modify the weights of the obstacle terms used in the cost function of the MPC shield 2a, in addition to adjusting the weights for matching the actions from the task agent, see Fig. 1. We adapt the Markov decision process (MDP), which is described by a tuple M: (S, A, R, P, γ). Where S is the set of states, A is the set of actions, $R(s, a)$ is the reward function, $P(s'|s, a)$ is the transition probability, and γ is the discount factor. An agent in state $s \in S$ takes an action $a \in A$ resulting in the next state $s' \in S$, which is rewarded by reward r and discounted by factor γ. The action a is chosen according to a policy π

that determines for each state which action the agent will take. The transition from state s to state s' upon taking action a is determined by the transition probability P.

The **observation space** for the supervisor agent includes the lidar data, the action from the task agent, the previous action from the MPC shield, and the relative angles and distances of the closest M obstacles fed to the MPC, which empirically improves performance. The **action space** of the agent includes the two weights for penalizing the deviation between the task agent and MPC shield, in addition to the M-dimensional weights corresponding to the obstacles considered by the MPC. We formulate the **reward function** as follows:

$$r = \begin{cases} -r_{collision} & \text{if collision or stuck,} \\ -min\_dist_{obst} \cdot \|Action_{RL} - Action_{MPC}\|^2 & \text{otherwise} \end{cases} \tag{3}$$

Since the supervisor agent is not responsible for the task completion, we do not give rewards for reaching the goal. Instead, we give the agent a large penalty $r_{collision}$ if the robot collides or if the robot is not moving for several consecutive steps (stuck), or a continuous penalty as a function of the distance to the single nearest obstacle and the difference between the action of the task agent and the MPC shield. That is, the further the robot is from the obstacles, the higher the penalty is for not matching the action from the task agent. As the robot approaches an obstacle, the penalty decreases, prioritizing safety over action alignment. This encourages the supervisor agent to follow the task agent's actions when far from obstacles and adjust them near obstacles to prevent collisions.

3.4. Task Agent

The task agent is also a SAC agent, whose observation space includes the lidar data, the previously taken action by the robot (v_{MPC} and ω_{MPC}), and the relative angle and distance to the goal location. The action space consists of v_{RL} and ω_{RL}. The reward function is as follows:

$$r = \begin{cases} r_{goal} & \text{if goal is reached,} \\ (goal\_dist_{t-1} - goal\_dist_t) & \text{otherwise.} \end{cases} \tag{4}$$

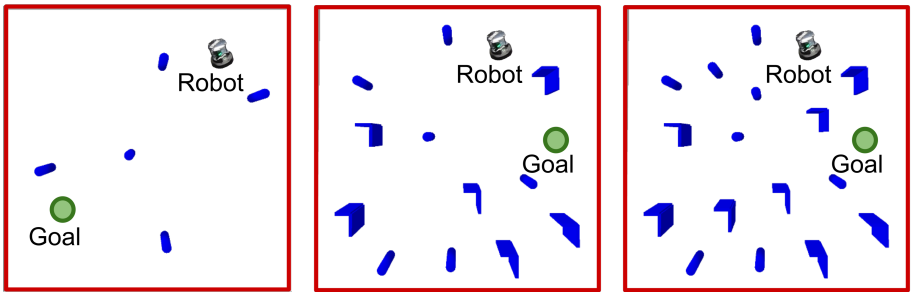
where the agent receives a large reward r_{goal} for reaching the goal and a continuous term that rewards the progress towards the goal. We do not penalize the collisions for this agent and rely on the supervisor instead to eliminate them.

4. Experiments

This work focuses on achieving safe reinforcement learning in navigation tasks by minimizing collisions without restricting the task agent's exploration. The experiments are designed to: (1) evaluate the impact of the dynamic shield on RL training, (2) compare the proposed approach with baseline methods in simulations, (3) analyze the effect of excluding task-dependent information for the supervisor, and (4) illustrate how the dynamic shield adjusts the constraints' weights on the real robot.

4.1. Baselines

For a fair comparison, we chose the baselines such that all of them are model-free RL, all baselines are trained from scratch and do not require pre-training, the baselines include variants of constrained-RL and safe exploration methods: **(i) SAC [Haarnoja et al. (2018)]**: Unconstrained



(*a*) Environment 1 (*b*) Environment 2 (*c*) Environment 3

Figure 2: Environments used in the experiments, Fig.**a** environment with five pillars (blue), and Fig.**b** environment, which contains six pillars and six L-shaped walls (blue). Fig.**c** environment with eight pillars and eight L-shaped walls. All the obstacles are placed randomly at the beginning of each episode.

SAC. **(ii) SAC-Lagrangian [Ray et al. (2019)] (SAC_LAG):** The Lagrangian variant of the SAC is a constrained-RL approach which uses two additional critic networks (Q_{cost}) to estimate the costs and optimizes the Lagrange multiplier online to balance the goal reaching with the rate of collisions. **(iii) SAC-PID [Stooke et al. (2020)] (SAC_PID)**[1]**:** A PID controller [Åström and Hägglund (2006)] is used along with the SAC-Lagrangian to update the Lagrange multiplier. The method has been shown to stabilize the training of the costs' critic (Q_{cost}) and has been commonly used as a baseline in safe RL studies. **(iv) MPC Safety Shield [Dawood et al. (2025)] (MPC_Tuned):** The MPC-based safety shield is a pre-tuned safety shield that ensures zero collisions in navigation scenarios. It has already been used in safe multi-agent RL and is classified as a safe exploration method.

4.2. Experimental Setup

We carry out the experiments in PyBullet [Coumans and Bai (2016–2021)] in three different environments. The first environment, Fig.2(*a*), contains five cylindrical obstacles (pillars) placed randomly in the environment at the beginning of each episode, the second environment consists of six pillars and six L-shaped walls, Fig.2(*b*), while the third consists of eight pillars and eight L-shaped walls, Fig.2(*c*). We chose the pillars as they are commonly used in obstacle avoidance scenarios, and the L-shaped walls as they represent challenging local minima during trajectory planning. During training, the RL agents, the obstacles locations, goals, and the robot location are randomized at the beginning of each episode. We apply episodic training, where the episode terminates if the robot collides or if it is not moving for 30 consecutive steps, or the maximum number of steps is reached. The robot is capable of reaching multiple goals within a single episode; a new goal is randomly generated each time the previous one is reached.

To train the supervisor agent while minimizing the number of collisions, we found it beneficial to repeatedly add samples involving collisions to its corresponding replay buffer. Empirically, duplicating each such sample three times provided the best results. Additionally, we train the supervisor agent more frequently compared to the task agent. Having different agents with separate replay buffers, makes it possible to train the agents for different number of steps without destabilizing the training. Finally, we found that annealing the learning rates for the supervisor agent is also extremely beneficial for achieving less number of collisions.

All the agents have been trained for one million steps from scratch. To assess the performance of the agents, we use three metrics as follows: **(i) Accumulated Number of Goals-to-Accumulated**

1. We used the same code as in [Jiaming Ji et al. (2023)] for SAC_LAG and SAC_PID

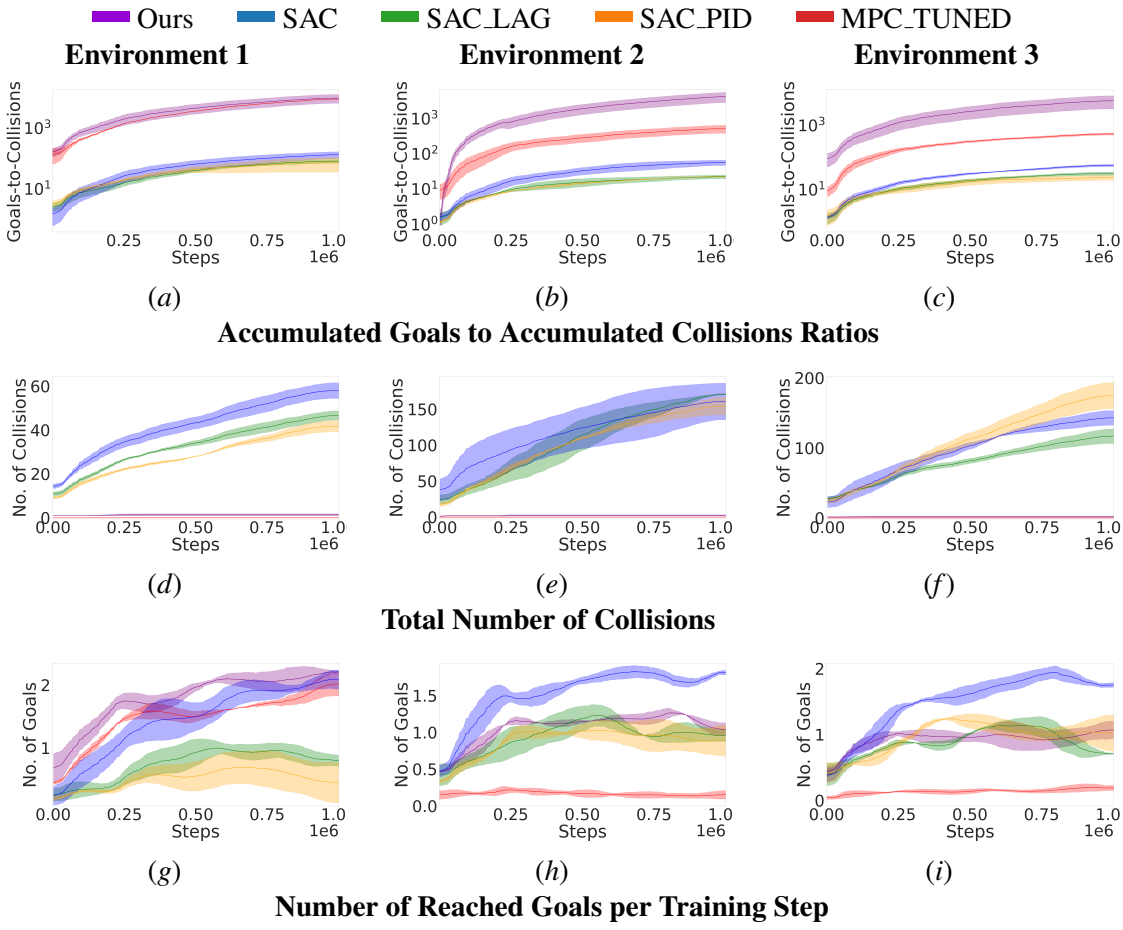
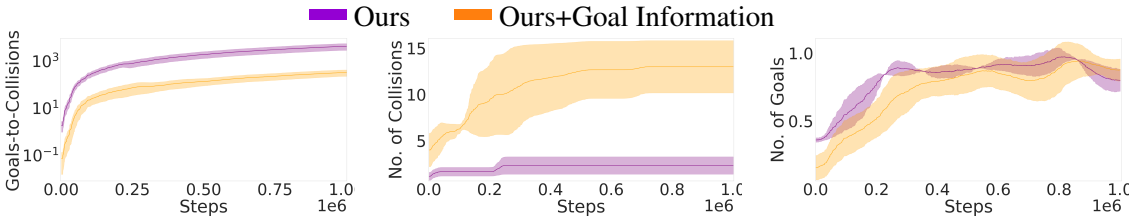


Figure 3: Results for all the approaches in the three environments. The bold lines show the average of three random seeds, while the shaded areas show the standard deviation over the runs. Our approach consistently achieves the highest goals-to-collisions ratio, which shows that our approach is able to safely guide the task agent to learn the navigation task with a few number of collisions. Note that some baselines reach up to 200 collisions, while our approach achieves near-zero collisions, overlapping with the MPC_TUNED at zero, as can be seen in the second row.

Number of Collisions Ratio: This metric has been introduced in [Thananjeyan et al. (2021)] to assess the performance of safe RL approaches as it signifies the importance of achieving more goals and less collisions. A higher ratio indicates fewer collisions while achieving more goals, which is desirable. In case of zero collisions, we set the number of collisions to one, to avoid division by zero. **(ii) Accumulated Number of Collisions:** Additionally, we count the total number of collisions during the training, as this is the main focus of safe RL. Fewer collisions indicate better safety adherence. **(iii) Number of Reached Goals:** Finally, we keep track of the number of goals reached at each training step to indicate the progress of the task agent. A higher number indicates better behavior from the task agent.

4.3. Results

Figure 3 shows the results of the three metrics for all the methods in the three environments. The figures show the means (bold lines) and standard deviations (shaded areas) over three runs for each approach. Figures 3(*a*), 3(*b*), and 3(*c*) illustrate that our approach achieves the highest



(*a*) Goals to Collisions Ratio (*b*) Total Number of Collisions (*c*) Number of Reached Goals

Figure 4: Results for the ablation study over three random seeds. Introducing the goal information to the supervisor agent, results in more collisions as the supervisor agent explores to reach more goals to maximize its rewards. The goals-to-collisions ratio for our approach without the goal information is higher than for the agent with the goal information. This shows that including the goal information for the supervisor agent does not improve the performance of the task agent.

goals-to-collisions ratio in all the environments compared to the baselines. In the first environment, MPC_TUNED has a high ratio as the MPC shield is able to eliminate the collisions completely while navigation to the goals. In the more challenging environments, the MPC_TUNED still maintains zero collisions, see Fig. 3(*d*), 3(*e*), and 3(*f*) but at the cost of over constraining the task agent resulting in the least number of goals reached among all the baselines. The unconstrained SAC is able to reach the highest number of goals, Fig. 3(*g*), 3(*h*), and 3(*i*) in all environments at the cost of large number of collisions, resulting in low accumulated goals-to accumulated collisions ratios. The constrained RL approaches SAC_LAG and SAC_PID on the other hand are able to decrease the number of collisions compared to the SAC at the cost of less goals reached, consequently leading to low ratios as well. These results demonstrate our method's ability to minimize collisions without over constraining the task agent.

4.4. Ablation Study

As mentioned previously, we do not provide any task information to the supervisor agent to minimize the exploration of the supervisor agent and hence minimize the number of collisions. In this section, we perform an ablation study in the first environment, where we add the goal information to the supervisor agent and give it a reward for reaching the goal and compare our approach without the goal information against the agent with the goal information. As can be seen from Fig. 4, introducing the goal information increases the number of collisions, as the added information expands the state space for the supervisor agent, resulting in a slower training process. This, in turn, reduces the goals-to-collisions ratio compared to our approach without the additional information. Nevertheless, the number of collisions remained lower than the constrained RL baselines.

4.5. Real-World Experiment

Finally, we investigate how the supervisor agent modifies the weights for the obstacles in a real-world experiment. Using checkpoints from our trained policies for both the task agent and the supervisor agent, we deployed these policies on a real robot. The policies were originally trained in simulation at 5 Hz, so we maintained the same control frequency on the robot to ensure consistency. We used the ROSbot XL [2] from Husarion for the experiments. The robot is equipped with a 360° lidar and an active tracker to track its location. See Fig. 5.

2. https://husarion.com/manuals/rosbot-xl/overview/

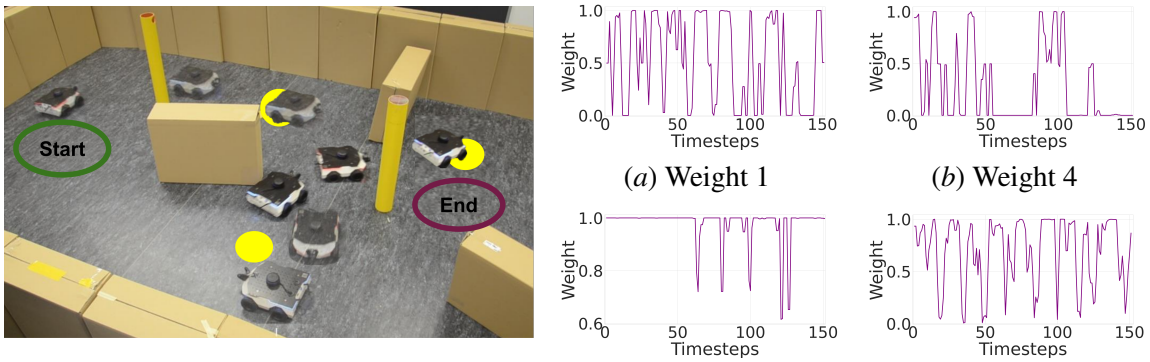


Figure 5: The figure illustrates the real-robot trajectory (left) and the weights adjusted by the supervisor agent (right). The robot navigates from the start position to the goals (yellow circles) while avoiding obstacles. The weight plots are arranged such that each weight corresponds to its respective quadrant in the lidar data. It can be seen that the supervisor agent increases the weights of the obstacles as the robot moves closer to them, prioritizing obstacle avoidance in the MPC shield.

The supervisor agent, implemented as an RL policy, dynamically adjusts the weights of the obstacles based on the current environment and state information. These weights influence the importance assigned to specific obstacles when calculating safe actions. For instance, as the robot moves closer to an obstacle, the supervisor agent increases the corresponding weight, prioritizing obstacle avoidance in the MPC shield. Conversely, weights for farther or less relevant obstacles may decrease, allowing the robot to focus on reaching its goal efficiently.

Fig.5 illustrates the robot's trajectory as it navigates toward marked goals while avoiding obstacles. The plot shows the real-time changes in weights as modified by the supervisor agent. These changes reflect the RL agent's decision-making process to balance obstacle avoidance with the task agent's goal-directed actions.

5. Conclusion

This work presents a novel safety shield approach for reinforcement learning (RL) in navigation tasks, designed to effectively balance safety and exploration. By combining the robustness of model predictive control (MPC) safety shields with the long predictive capabilities of RL, we introduced a dynamic system where a supervisor agent adjusts the weights for the obstacle avoidance terms and for aligning the MPC actions with the task agent's actions.

Our experiments demonstrate the superiority of the proposed approach across diverse environments. The results show that our method achieves the highest goals-to-collisions ratio, significantly minimizing collisions compared to constrained RL methods, i.e., Stooke et al. (2020); Ray et al. (2019). Unlike classical MPC-based shields Dawood et al. (2025), which often over-constrains the RL agents, our method promotes exploration without compromising safety, enabling the task agent to achieve higher rewards in challenging scenarios. Moreover, compared to unconstrained RL methods, which maximize goals at the cost of collisions, our approach strikes a balance by reducing collisions while maintaining competitive performance in goal-reaching metrics.

The effectiveness of the dynamic adjustment mechanism was evident in its ability to adapt to varying environmental complexities, ensuring fewer constraints violations without hindering the task agent's progress. This balance emphasizes the potential of integrating optimization-based methods with RL to address real-world challenges in safe exploration.

Acknowledgments

Murad Dawood, Ahmed Shokry, and Maren Bennwitz are with the Humanoid Robots Lab, University of Bonn, Germany and with the Lamarr Institute for Machine Learning and Artificial Intelligence as well as the Center for Robotics, Bonn, Germany. This work has been partially funded by the BMBF within the Robotics Institute Germany, grant No. 16ME0999.

References

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pages 22–31. PMLR, 2017.

Ahmed Agha, Baris Kayalibay, Atanas Mirchev, Patrick van der Smagt, and Justin Bayer. Exploring under constraints with model-based actor-critic and safety filters. In *8th Annual Conference on Robot Learning*, 2024.

Aaron D Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *2019 18th European control conference (ECC)*, pages 3420–3431. IEEE, 2019.

Karl Johan Åström and Tore Hägglund. *Advanced PID control*. ISA-The Instrumentation, Systems and Automation Society, 2006.

Bruno Brito, Michael Everett, Jonathan P How, and Javier Alonso-Mora. Where to go next: learning a subgoal recommendation policy for navigation in dynamic environments. *IEEE Robotics and Automation Letters (RA-L)*, 2021.

Steven Carr, Nils Jansen, Sebastian Junges, and Ufuk Topcu. Safe reinforcement learning via shielding under partial observability. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 14748–14756, 2023.

Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2021.

Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757*, 2018.

Murad Dawood, Sicong Pan, Nils Dengler, Siqi Zhou, Angela P Schoellig, and Maren Bennewitz. Safe multi-agent reinforcement learning for behavior-based cooperative navigation. *arXiv preprint arXiv:2312.12861*, 2025.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

Tairan He, Chong Zhang, Wenli Xiao, Guanqi He, Changliu Liu, and Guanya Shi. Agile but safe: Learning collision-free high-speed legged locomotion. *arXiv preprint arXiv:2401.17583*, 2024.

Weidong Huang, Jiaming Ji, Borong Zhang, Chunhe Xia, and Yaodong Yang. Safe dreamerv3: Safe reinforcement learning with world models. *arXiv preprint arXiv:2307.07176*, 2023.

Jiaming Ji, Jiayi Zhou, Borong Zhang, Juntao Dai, Xuehai Pan, Ruiyang Sun, Weidong Huang, Yiran Geng, Mickel Liu, and Yaodong Yang. Omnisafe: An infrastructure for accelerating safe reinforcement learning research. *arXiv preprint arXiv:2305.09304*, 2023.

Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. *arXiv preprint arXiv:1910.01708*, 7(1):2, 2019.

Aivar Sootla, Alexander I Cowen-Rivers, Taher Jafferjee, Ziyan Wang, David H Mguni, Jun Wang, and Haitham Ammar. Sauté rl: Almost surely safe reinforcement learning using state augmentation. In *International Conference on Machine Learning*, pages 20423–20443. PMLR, 2022.

Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by pid lagrangian methods. In *International Conference on Machine Learning*, pages 9133–9143. PMLR, 2020.

Chen Tessler, Daniel J Mankowitz, and Shie Mannor. Reward constrained policy optimization. *arXiv preprint arXiv:1805.11074*, 2018.

Brijen Thananjeyan, Ashwin Balakrishna, Suraj Nair, Michael Luo, Krishnan Srinivasan, Minho Hwang, Joseph E Gonzalez, Julian Ibarz, Chelsea Finn, and Ken Goldberg. Recovery rl: Safe reinforcement learning with learned recovery zones. *IEEE Robotics and Automation Letters*, 6 (3):4915–4922, 2021.

Kim P Wabersich, Lukas Hewing, Andrea Carron, and Melanie N Zeilinger. Probabilistic model predictive safety certification for learning-based control. *IEEE Transactions on Automatic Control*, 67(1):176–188, 2021.

Kim Peter Wabersich and Melanie N Zeilinger. A predictive safety filter for learning-based control of constrained nonlinear dynamical systems. *Automatica*, 129:109597, 2021.

Zifan Xu, Bo Liu, Xuesu Xiao, Anirudh Nair, and Peter Stone. Benchmarking reinforcement learning techniques for autonomous navigation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9224–9230. IEEE, 2023.

Wenbo Zhang, Osbert Bastani, and Vijay Kumar. Mamps: Safe multi-agent reinforcement learning via model predictive shielding. *arXiv preprint arXiv:1910.12639*, 2019.

Zhili Zhang, Songyang Han, Jiangwei Wang, and Fei Miao. Spatial-temporal-aware safe multi-agent reinforcement learning of connected autonomous vehicles in challenging scenarios. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5574–5580. IEEE, 2023.