Policy-Guided Model Predictive Path Integral for Safe Manipulator Trajectory Planning with Constrained Discounted Reinforcement Learning

Liang Liang *, Chengdong Wu, Xiaofeng Wang

Posted Date: 3 March 2026

doi: 10.20944/preprints202603.0086.v1

Keywords: manipulator trajectory planning; model predictive path integral; reinforcement learning; control barrier function; configuration-space distance field

 doi:10.20944/preprints202603.0086.v1

Article

Policy-Guided Model Predictive Path Integral for Safe Manipulator Trajectory Planning with Constrained Discounted Reinforcement Learning

Liang Liang [1,2,*], Chengdong Wu [3] and Xiaofeng Wang [4]

[1] College of Information Science and Engineering, Northeastern University, Shenyang 110819, China
[2] SIASUN Robot & Automation Co., Ltd., Shenyang 110168, China
[3] Faculty of Robot Science and Engineering, Northeastern University, Shenyang 110169, China
[4] SIASUN Robot & Automation Co., Ltd., Shenyang 110168, China
* Correspondence: liangliang@siasun.com

What are the main findings?

- The proposed PG-MPPI framework integrates CD-SAC offline learning, MPPI online planning and CBF safety filtering to construct a three-level safety system, which addresses the challenges of difficult hard constraint enforcement and weak environmental generalization in manipulator trajectory planning.
- The algorithm achieves a 100% success rate of collision-free target reaching in multi-scenario simulations on the SIASUN T12B manipulator, with the trajectory satisfying physical constraints and demonstrating excellent adaptability.

What are the implications of the main findings?

- The algorithm provides a new paradigm integrating global prior learning and local real-time optimization for manipulator trajectory planning, breaking through the limitations of traditional methods in sampling efficiency and constraint handling.
- Its multi-level safety assurance mechanism can be transferred to practical industrial scenarios, providing technical support for the application of industrial robots in complex scenarios such as flexible manufacturing and human-robot collaboration.

Abstract

Aiming at the problems of difficult hard constraint enforcement, weak environmental generalization ability in the safe trajectory planning of manipulators in complex environments, a Policy-Guided Model Predictive Path Integral (PG-MPPI) planning framework is proposed. This framework integrates the advantages of reinforcement learning and model predictive control to construct a global prior guidance, local real-time optimization and hard constraint safety assurance: a Constraint-Discounted Soft Actor-Critic (CD-SAC) offline learning policy is designed, which incorporates the configuration-space distance field as a safety guidance term to realize the learning of obstacle avoidance behavior; the offline policy is used to guide the online sampling and optimization of MPPI, improving sampling efficiency and planning quality; a Control Barrier Function (CBF) safety filter is introduced to revise control commands in real time, ensuring the strict satisfaction of constraints. Taking the SIASUN T12B manipulator as the research object, simulation comparison experiments are carried out in multi-obstacle scenarios. The results show that the PG-MPPI algorithm outperforms the comparison algorithms in the success rate of collision-free target reaching, ensure the smoothness and feasibility of the trajectory, and has a good adaptive capacity to dynamic environments, thus providing an efficient solution for the autonomous and safe operation of manipulators.

Keywords: manipulator trajectory planning; model predictive path integral; reinforcement learning; control barrier function; configuration-space distance field

 doi:10.20944/preprints202603.0086.v1

1. Introduction

With the development of industrial automation towards flexible manufacturing and human-robot collaboration, manipulators are required to perform tasks in highly unstructured and dynamically changing environments [1]. This demands that motion planning algorithms not only have real-time obstacle avoidance capability, but also ensure trajectory smoothness and time optimality while satisfying complex constraints [2]. Traditional sampling-based methods such as RRT* and PRM can guarantee probabilistic completeness, yet suffer from low search efficiency in high-dimensional spaces and fail to satisfy the millisecond-level requirement for online re-planning [3]. Optimization-based methods like CHOMP and TrajOpt yield high-quality trajectories, but are prone to falling into local minima and have limited ability to handle non-convex obstacles [4].

Model Predictive Control (MPC) has emerged as a powerful tool to address this problem due to its capability in handling multivariable constraints and receding horizon optimization [5]. In particular, the sampling-based Model Predictive Path Integral (MPPI) leverages the large-scale parallel computing capability of Graphics Processing Units (GPUs) to effectively handle non-smooth and non-convex cost functions by applying random perturbations to control sequences and performing weighted averaging [6]. Compared with gradient-based MPC, MPPI is not restricted by the differentiability of cost functions and exhibits stronger robustness [7]. However, the performance of standard MPPI is highly dependent on the quality of the nominal control sequence and coverage of the sampling distribution. When confronted with long-horizon planning or complex geometric traps in the environment (e.g., U-shaped obstacles or narrow passages), simple mean initialization often leads to extremely low sampling efficiency, and the algorithm is likely to converge to suboptimal solutions or even fail to find feasible ones [8]. In addition, how to strictly satisfy hard safety constraints while ensuring sampling diversity that remains a major challenge for MPPI [9].

Deep reinforcement learning (RL) can learn policies and value functions with a global perspective through offline training and features extremely fast inference speed [10]. Nevertheless, pure end-to-end RL policies often lack adaptability to environmental changes and fail to provide rigorous hard safety guarantees [11]. To combine the accuracy of planning algorithms with the speed of learning algorithms, recent studies have begun to explore hybrid architectures of RL and MPC. A mainstream approach is to use the learned policy as a warm start for MPC to guide the search process [12,13]. For example, TD-MPC integrates model learning and planning in the latent space [14]; RL-Driven MPPI directly takes the trajectory generated by the offline policy as the mean of MPPI, which significantly accelerates the convergence speed of online planning [8]. Despite this, existing hybrid methods still adopt simplified Signed Distance Functions (SDF) [15] or sparse collision penalties when handling the configuration-space safety of high-dimensional manipulators, ignoring the complexity of the topological structure of the joint space.

In terms of safety constraint handling, Control Barrier Functions (CBF) provide a theoretically rigorous safety assurance mechanism [16,17]. By defining the safe set as an invariant set of the state space, CBF can act as a filter to revise the nominal control law. However, the construction of CBF often relies on accurate models and gradients, and its design in high-dimensional non-convex obstacle environments is extremely challenging. For geometric obstacle avoidance, the Configuration-space Distance Field (CDF) can directly reflect the proximity between the overall configuration of the manipulator and obstacles, in contrast to the traditional workspace SDF [18,19]. Introducing CDF into the reward shaping of RL or the cost function of MPC can provide a smoother and globally guided safety gradient [19]. In addition, to solve the oscillation problem caused by soft constraints, the Constraint Discounting (CD) mechanism is proposed [20], which achieves soft termination by dynamically adjusting the discount factor of future rewards and exhibits superior stability in safety learning.

Based on the above background, this paper proposes a Policy-Guided Model Predictive Path Integral (PG-MPPI) control framework aiming to solve the problem of safe motion planning for high-dimensional manipulators in complex dynamic environments. The main contributions of this paper are as follows:

 doi:10.20944/preprints202603.0086.v1

1. A dense safety-guided policy learning method based on CDF is proposed. Breaking through the limitation of sparse collision penalties of traditional workspace signed distance functions, CDF is integrated into the design of the SAC reward function, and its continuously differentiable gradient characteristic is utilized to provide global dense guidance for obstacle avoidance behavior;
2. A policy-guided MPPI online planning architecture is designed to achieve deep integration of global prior and local optimization. The offline-trained Constraint-Discounted SAC (CD-SAC) policy is used as the nominal control sequence generator for MPPI to provide a high-quality warm start, concentrating the sampling distribution near the global optimal solution. It resolves traditional MPPI's drawbacks, and corrects the offline policy's generalization error via MPPI's online receding horizon optimization;
3. A multi-level safety system featuring "policy soft guidance + optimization hard constraint + filter final guarantee" is constructed. At the offline learning layer, the safety preference is internalized into the policy through CDF and TD-CD; at the online planning layer, the trajectory feasibility is enhanced through the explicit cost penalty of MPPI; at the execution layer, a safety filter based on first-order CBF is introduced to perform real-time projection correction on control commands.

The rest of this paper is organized as follows: Section 2 presents the problem formulation and relevant preliminaries. Section 3 elaborates on the implementation details of the proposed PG-MPPI algorithm, including the design of the offline constraint-discounted reinforcement learning policy and the online planning system. Section 4 verifies the effectiveness and superiority of the algorithm through multi-scenario simulation experiments. Section 5 summarizes the overall work of this paper and prospects future research directions.

2. Preliminaries

2.1. Problem Formulation

For the trajectory planning problem of an n-degree-of-freedom manipulator, we define $q \in \mathbb{R}^n$ as the joint angle vector in the robot's configuration space. The joint angles and joint velocities form the system state vector $x = [q, \dot{q}]$, and the joint acceleration $\ddot{q}$ is taken as the control input vector u. The manipulator's dynamic system is modeled in discrete-time form:

$$x_{t+1} = f(x_t, u_t), \tag{1}$$

where $f(\cdot)$ denotes the discrete dynamic function , and t represents the time step.

The core objective of this study is to seek a sequence of control inputs over an infinite time horizon to minimize the cumulative cost:

$$\min_{\{u_0, u_1, \ldots\}} \sum_{t=0}^{\infty} \ell(x_t, u_t), \tag{2}$$

where $\ell(\cdot)$ denotes the stage cost function.

To ensure the safety, physical feasibility and motion smoothness of the manipulator's trajectory, three types of strict core constraints are further introduced on the basis of the aforementioned optimal control objective, forming a constrained optimal control problem:

Obstacle avoidance constraint: Define the minimum safety distance d_{safe} between the manipulator's links, joints and obstacles in the workspace, and require the actual minimum distance to satisfy the safety threshold $d(x_t) \geq d_{\text{safe}}$ during motion to ensure no geometric collision;

4. State constraint: Set upper and lower bounds for joint angles and joint velocities, i.e., $q_{\min} \leq q \leq q_{\max}$, $\dot{q}_{\min} \leq \dot{q} \leq \dot{q}_{\max}$, to prevent the system from entering a physically infeasible state for the mechanical structure and drive system;

 doi:10.20944/preprints202603.0086.v1

5. Control constraint: Set an amplitude upper bound for joint acceleration, i.e., $\|u\| \le u_{\max}$, to match the actual output capability of actuators and prevent mechanical vibration, impact or hardware damage caused by abrupt changes in control signals.

The above constrained infinite-horizon optimal control model constitutes the basic mathematical framework for safe trajectory planning of manipulators.

2.2. Model Predictive Path Integral

MPPI is a class of sampling-based model predictive control methods [6]. Its core idea is to perform random sampling and search around the nominal control sequence, abandoning the traditional approach of selecting a single optimal trajectory. This method is suitable for trajectory planning scenarios of nonlinear and high-dimensional systems.

For the initial nominal control sequence $\bar{U}=\{\bar{u}_0,...,\bar{u}_H\}$ within the receding horizon H, an independent zero-mean Gaussian noise sequence is generated for each of the K sampled trajectories at time t:

$$\varepsilon_i^k \sim N(0,\Sigma), \tag{3}$$

where $i=0,...,H-1$, $k=1,...,K$, Σ denote the noise covariance matrix. Based on this, the candidate control sequence for the k-th trajectory is constructed as:

$$u_i^k = \bar{u}_i + \varepsilon_i^k . \tag{4}$$

A forward rollout is executed for each candidate control sequence, and the corresponding state trajectory $\{x_{t+i}^k\}_{t=0}^H$ is obtained by combining with (1). The cumulative cost over the receding horizon is given by:

$$S_k = S(U^k; x_t^k) = \sum_{i=0}^{H-1} \ell(x_{t+i}^k, u_{t+i}^k) + \ell_f(x_{t+H}^k), \tag{5}$$

where $U=\{u_t,...,u_{t+H-1}\}$ is the control sequence, $\ell(\cdot)$ is the stage cost function, and $\ell_f(\cdot)$ is the terminal cost function. An exponential transformation is applied to the cumulative cost of each sampled trajectory to construct an exponential weight:

$$\omega_k = \exp\left(-\frac{1}{\lambda}(S_k - S_{\min})\right), \tag{6}$$

where $\lambda > 0$ is the temperature parameter. Smaller value of λ results in a stronger bias of the weight toward low-cost trajectories, whereas a larger value enhances the exploratory nature of sampling. The normalized weights $\tilde{\omega}_k$ are used to perform a weighted average of all sampled noise sequences, thus realizing the update of the initial nominal control sequence:

$$\bar{u}_i \leftarrow u_i + \sum_{k=1}^{K} \tilde{\omega}_k \varepsilon_i^k . \tag{7}$$

MPPI adopts a receding horizon execution strategy, where only the control signal $u_t^* = \bar{u}_0$ at the initial time step of the updated nominal control sequence is extracted and applied to the manipulator's actuators to drive the robot to complete the motion of the current time step. When proceeding to the next time step t+1, a shifting operation needs to be performed on the control sequence.

2.3. Soft Actor-Critic

The core objective of reinforcement learning is to learn an approximately optimal policy π that enables an agent to maximize the long-term expected return during its interactions with the environment. Soft Actor-Critic (SAC) is an off-policy Actor-Critic algorithm based on the maximum

 doi:10.20944/preprints202603.0086.v1

entropy reinforcement learning framework [21], which maximizes both the expected cumulative return and policy entropy simultaneously, formulated as:

$$\max_{\pi} \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t\left(r\left(s_t,a_t\right)+\alpha\mathcal{H}\left(\pi\left(\cdot\mid s_t\right)\right)\right)\right], \tag{8}$$

where $\gamma\in[0,1)$ is the discount factor, and $r\left(s_t,a_t\right)$ is the immediate reward obtained by the agent executing action a_t in state s_t. A larger entropy value $\mathcal{H}\left(\pi\left(\cdot\mid s_t\right)\right)=-\mathbb{E}_{a\sim\pi(\cdot|s_t)}\left[\log\pi\left(a\mid s_t\right)\right]$ means the policy has stronger exploratory properties. $\alpha>0$ is the temperature parameter.

SAC adopts a twin Q-networks to approximate the state-action value function, so as to alleviate the value overestimation problem of the Q-networks. For samples $\left(s_t,a_t,r_t,s_{t+1},d_t\right)$ where d_t is the termination flag in the replay buffer $\mathcal{D}$, the soft Bellman backup target is defined as:

$$y_t=r_t+\gamma\left(1-d_t\right)\mathbb{E}_{a_{t+1}\sim\pi_\phi(\cdot|s_{t+1})}\left[\min_{i\in\{1,2\}}Q_{\bar{\theta}_i}\left(s_{t+1},a_{t+1}\right)-\alpha\log\pi_\phi\left(a_{t+1}\mid s_{t+1}\right)\right], \tag{9}$$

where ϕ denotes the learnable parameters of the actor network.

The training objective of the critic network is to minimize the mean squared error of the temporal difference (TD) error, given by:

$$\min_{\theta_i}\mathbb{E}_{(s_t,a_t,r_t,s_{t+1},d_t)\sim\mathcal{D}}\left[\left(Q_{\theta_i}\left(s_t,a_t\right)-y_t\right)^2\right], \tag{10}$$

where θ_i denotes the parameters of the target critic network.

The actor network adopts a stochastic policy formulation and outputs the probability distribution of all actions under a given state s. Its core training objective is to maximize the expected soft value function. The objective function for policy update is given by:

$$\min_{\phi}\mathbb{E}_{s_t\sim\mathcal{D},\ a_t\sim\pi_\phi}\left[\alpha\log\pi_\phi\left(a_t\mid s_t\right)-\min_{i\in\{1,2\}}Q_{\theta_i}\left(s_t,a_t\right)\right]. \tag{11}$$

In SAC, the temperature parameter α is treated as a learnable parameter and updated by maximizing the entropy-weighted expected return objective. Based on the aforementioned principles, SAC leverages neural networks to learn the optimal policy, which satisfies the maximization condition of the objective function (8).

3. Policy-Guided MPPI

As discussed in Section 2, RL learns a parameterized policy by solving Equation (8) offline using samples collected from interactions with the environment. The learned RL policy exhibits excellent real-time performance in the online decision-making process, yet may fail in certain states due to sub-optimality during training and rare state visits. In contrast, MPPI minimizes Equation (5) via rollouts starting from the current state to find the optimal control sequence at each time step. It outperforms the RL policy in ensuring policy feasibility across all feasible states, but results in poor online decision-making efficiency. In this section, we propose a method that combines RL with MPPI, thereby providing better control performance for complex systems with flexible cost criteria.

3.1. Algorithm Framework

The proposed PG-MPPI algorithm framework integrates the global policy guidance capability of reinforcement learning with the local online fine optimization capability of MPPI to realize the globally and locally coordinated safe trajectory planning of manipulators in complex environments. On the whole, it forms a closed-loop control system featuring global prior guidance, local real-time optimization and hard safety constraint guarantee. The algorithm architecture is shown in Figure 1.

 doi:10.20944/preprints202603.0086.v1

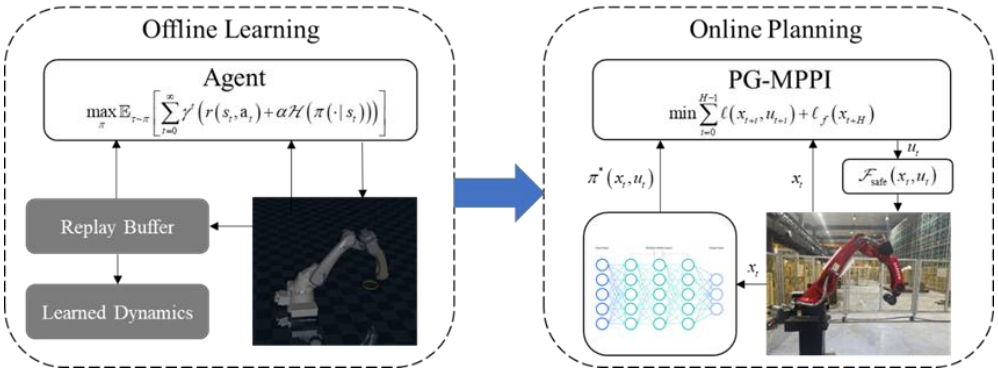

Figure 1. Architecture of PG-MPPI.

This framework is mainly divided into two modules: an offline learning phase and an online planning phase:

Offline Learning: A prior controller is obtained through repeated interactions of reinforcement learning in a simulation environment. The objectives of target reaching + safety constraints are encoded into the reward function or constraint discounting mechanism to train a policy $\pi_\phi(a \mid s)$. Such a policy is capable of providing action guidance with a long planning horizon and shifting a large amount of computational load from the online execution phase to the offline training phase;

Online Planning: Real-time optimization is performed under the current actual state and environment to ensure safe execution of the manipulator. The output of the offline policy is adopted as the nominal control sequence of MPPI; random sampling and rollout are conducted around this sequence. The desired control signal is then revised by a safety filter to satisfy all safety constraints. This approach not only leverages the RL prior to improve sampling efficiency, but also utilizes the explicit cost function and receding horizon optimization of MPPI.

3.2. Offline Policy Learning

With the precise positioning of the manipulator's end-effector at the designated target as the core primary task, the Configuration-space Distance Field (CDF) is adopted as the core safety guidance term. Its gradient information and a safety penalty mechanism are integrated into the design of the reinforcement learning reward function, guiding the policy to actively learn obstacle avoidance behaviors during offline training. This enables the manipulator to autonomously keep away from obstacle collision areas, achieving the collaborative optimization of task completion and safe obstacle avoidance.

3.2.1. Reward Design

Let the Euclidean distance between the manipulator's end-effector $x(q_t)$ and the target position x_g be expressed as:

$$d_t = \left\| x(q_t) - x_g \right\|_2 . \tag{12}$$

Based on this distance, a progress-based target reward is designed to make the reward signal change continuously as the end-effector approaches the target, providing a stable gradient guidance for policy optimization, with the form:

$$r_{\text{goal}}(t) = k_g \left(d_{t-1} - d_t \right), \tag{13}$$

where k_g denotes the reward weight, which is used to adjust the proportion of the primary task in the reward function.

 doi:10.20944/preprints202603.0086.v1

Traditional obstacle modeling usually employs the SDF f_s to represent the distance between a point p in space and the robot's surface $\partial r(q)$:

$$f_s(p,q) = \pm \min_{p' \in \partial r(q)} \|p - p'\|, \tag{14}$$

where $\pm$ is a sign indicating the distance direction, which is positive if the point is outside and negative if inside. Such models only treat collisions as hard constraints or simple soft penalties, increasing the cost when a collision is detected. This approach fails to fully utilize the obstacle distance information to provide active guidance for collision-free path planning, thus leading to low learning efficiency of obstacle avoidance policies.

To address the above problems, this study adopts the Configuration-space Distance Field (CDF) for obstacle modeling [19]. Taking joint values as independent variables, this function represents the minimum distance from the manipulator's current joint configuration to the critical joint configuration where a collision occurs, and this distance determines the contact between the robot and the obstacle:

$$f_c(p,q) = \min_{q'} \|q - q'\|, \tag{15}$$

where q' satisfies collision constraints , and the configuration set corresponding to a given point p can be found on the zero-level set of the SDF model via inverse kinematics.

The CDF is continuous and differentiable almost everywhere in the configuration space, and its gradient naturally points to the locally optimal collision escape direction. It can provide a direct gradient guidance signal for obstacle avoidance behavior learning, making it an ideal safety guidance term for manipulator joint space trajectory planning. The principle is shown in Figure 2.

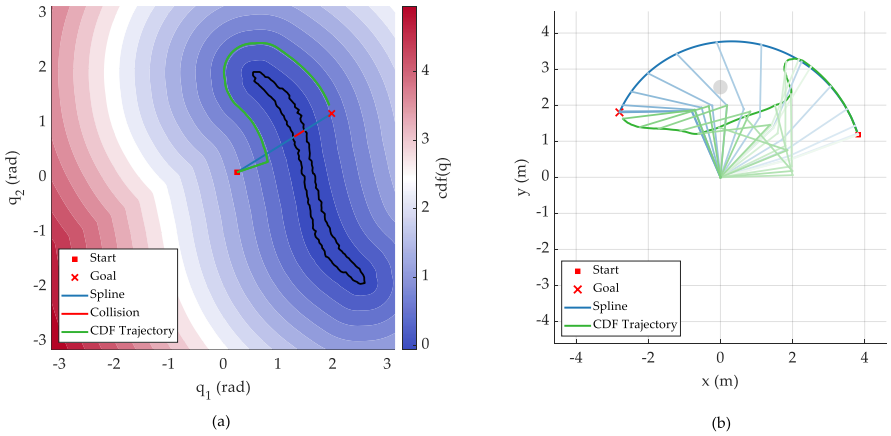

Figure 2. Schematic of CDF-based 2-DOF trajectory. (**a**) CDF contour lines and trajectory in the configuration space. The black contour line is the set of configurations in contact with obstacles. Motion along the tangent and gradient of the contour lines can bypass obstacles; (**b**) Workspace trajectory corresponding to the configuration space trajectory.

A nonlinear safety penalty term is designed based on the CDF to realize adaptive punishment for collision risks, with the form:

$$r_{\text{safe}}(t) = -\mu \left[\max\left(0, d_{\text{m}} - f_c(q_t)\right) \right]^p, \tag{16}$$

where $\mu > 0$ is the safety penalty weight, $p \geq 1$ is the penalty exponent, and $d_{\text{m}} > 0$ is the safety margin threshold. When the manipulator's joint configuration is in the safe region $f_c(q_t) \geq d_{\text{m}}$, the safety penalty term is 0 and does not interfere with the execution of the primary task. When the joint configuration enters the low-CDF risk region $f_c(q_t) < d_{\text{m}}$, the penalty value increases nonlinearly

 doi:10.20944/preprints202603.0086.v1

with the decrease of the CDF value. A larger penalty exponent results in a higher penalty intensity in the collision risk region, which can quickly push the policy back to the safe region while avoiding excessive punishment for minor boundary crossings.

To further strengthen safety constraints, the current training episode is terminated immediately and a one-time collision penalty is imposed when an actual collision occurs to the manipulator. Combining the target reward, safety penalty term and one-time collision penalty, the total reward function is obtained as:

$$r_t = r_{\text{goal}}(t) + r_{\text{safe}}(t) + r_{\text{col}}(t). \tag{17}$$

This reward function achieves an organic integration of the primary task objective and safety obstacle avoidance constraints. It not only drives the manipulator to complete the end-effector positioning task through the progress-based reward, but also enables the policy to actively learn obstacle avoidance behaviors during offline training by virtue of the CDF's gradient guidance and nonlinear penalty mechanism, thus ensuring the task feasibility and collision-free safety of the trajectory from the perspective of reward design.

3.2.2. Constraints Based on Discount Factor

To achieve unified and adaptive management of multiple types of constraints in manipulator trajectory planning, this section converts the degree of constraint violation into a reward discount weight through a random termination signal, and combines the Exponential Moving Average (EMA) and finite-horizon discount rules to form a constraint handling scheme that balances safety and control continuity.

Various physical and safety constraints during the manipulator's motion are uniformly abstracted into a general form of inequality constraints, covering all hard constraints such as joint position boundaries and the amplitude limits of joint velocity and acceleration:

$$c_i(x,u) \le 0 \quad \forall i \in \mathcal{I}. \tag{18}$$

To avoid the threshold dependence and risks of constraint violation superposition associated with manually designed penalty functions, this section adopts a random termination signal based on constraint violation to directly map the degree of violation to a discount weight for future rewards, which is defined as:

$$\delta_t = \max_{i \in \mathcal{I}} \left(p_i^{\max} \cdot \text{clip}\left(\frac{[c_t]_i}{c_i^{\max}}, 0, 1 \right) \right), \tag{19}$$

where $p_i^{\max} \in [0,1]$ is the maximum termination probability parameter for *i*-th constraint, $[c_t]_i$ is the actual violation amount of constraint *i* at step *t*, and $c_i^{\max}$ is the maximum violation value of the *i*-th constraint across all trajectory samples. Equation (19) has the following properties:

Constraint violation no longer relies on manually designed penalty functions. Instead, the degree of violation is directly converted into a discount weight for future rewards. The more severe the violation, the closer the value δ_t is to $p_i^{\max}$, and the higher the proportion of future rewards being discounted;

A high termination probability is triggered as long as one constraint is severely violated, which prevents safety risks or motion failure caused by the superposition of multiple minor violations and ensures the stringency of constraints;

In the absence of violations, $[c_t]_i = 0$ and $\delta_t = 0$, meaning future rewards are not discounted to encourage normal exploration. In the case of minor violations, $0 < [c_t]_i < c_t^{\max}$ and $\delta_t \in (0, p_i^{\max})$, future rewards are partially discounted, which not only warns of violations but also allows the robot

 doi:10.20944/preprints202603.0086.v1

to learn recovery strategies. In the case of severe violations, $[c_t]_i \geq c_t^{\max}$ and $\delta_t = p_i^{\max}$, which directly terminates all future rewards and strictly prohibits severe violations.

To avoid fluctuations in the discount factor caused by extreme violation values in a single training batch, and to realize data-driven adaptive learning of the violation reference range, an EMA mechanism is adopted to update the historical maximum violation value of each constraint $c_i^{\max}$:

$$c_i^{\max} \leftarrow \tau_c \cdot \hat{c}_i^{\max} + (1-\tau_c) \cdot c_i^{\max}, \tag{20}$$

where $\hat{c}_i^{\max} = \max_t [c_t]_i$ is the maximum violation value of the *i*-th constraint at all time steps in the current training batch, and $\tau_c \in [0,1]$ is the decay rate parameter representing the percentage of historical information retained. This mechanism enables the violation reference value to be dynamically adjusted with the training process, quickly adapting to the actual range of environmental constraints in the early stage and tending to be stable in the later stage.

3.2.3. Constraint-Discounted SAC Learning Strategy

Based on the aforementioned constraint discounting mechanism and combined with the maximum entropy reinforcement learning framework of the SAC algorithm, CD-SAC learning strategy is adopted. Constraint information is injected into the temporal-difference backup process of SAC in the form of soft termination, enabling multi-objective fusion learning for primary task reward, safety guidance, and constraint discounting.

The agent performs rollout sampling for state transitions (s_t, u_t, r_t, s_{t+1}) in a constrained dynamic environment, and maps the degree of constraint violation at each step to a soft termination intensity δ_t according to Equation (19), thereby deriving a time-varying discount factor γ_t:

$$\gamma_t = \gamma\left(1 - \text{clip}(\delta_t, 0, 1)\right), \tag{21}$$

where γ denotes the baseline discount factor of SAC. When a constraint violation occurs or is imminent, γ_t is reduced, thus weakening the propagation of future rewards across unsafe state transitions and achieving soft termination.

In the critic network update, the twin Q-networks structure and entropy regularization form of SAC remain unchanged, with only the constant discount factor in the temporal-difference (TD) target replaced by the time-varying γ_t. For mini-batches $\{(s_t, u_t, r_t, s_{t+1}, \gamma_t)\}$ sampled from the replay buffer, a soft value function target is constructed as follows:

$$V(s_{t+1}) = \mathbb{E}_{a_{t+1} \sim \pi(\cdot|s_{t+1})}\left[\min\left(Q_{\bar{\theta}_1}(s_{t+1}, a), Q_{\bar{\theta}_2}(s_{t+1}, a)\right) - \alpha \log \pi(a \mid s_{t+1})\right]. \tag{22}$$

The Temporal-Difference Constraint-Discounting (TD-CD) target is then given by:

$$y_t = r_t + \gamma_t V(s_{t+1}). \tag{23}$$

The update rules for the critic network, actor network and temperature parameter all follow the standard SAC formulation, and the overall training process remains an online interactive off-policy learning paradigm. The only difference is that the replay buffer additionally stores the γ_t value for each state transition, thus injecting constraint information into the TD backup of the critic network.

In contrast to hard reward penalties for constraint violations, TD-CD exerts the effect of constraints on the value propagation path: the discount factor is automatically reduced when constraint violations occur or their severity increases, which attenuates the impact of subsequent future rewards on the current state-action value and mitigates the overestimation of value and gradient amplification by the policy in unsafe regions. This mechanism does not rely on on-policy trajectories and is naturally compatible with experience replay buffers, resulting in smoother and more stable training.

 doi:10.20944/preprints202603.0086.v1

3.3. Online Trajectory Planning

The offline-trained CD-SAC policy can provide global and long-horizon action guidance for manipulator planning and avoid static obstacles in the training scenarios, but it lacks the local adaptability to unmodeled dynamic obstacles. To address this issue, this paper fuses this offline policy with the local online optimization capability of MPPI and introduces a CBF safety filter.

3.3.1. CBF-Based Safety Filter

The safety filter serves as the final safety barrier in the online planning phase, whose core function is to revise the desired control commands output by MPPI in real time. This ensures that the final control signals applied to the manipulator strictly satisfy the hard constraints of obstacle avoidance, state and control, thus guaranteeing the system's safety from a theoretical perspective.

Taking the first-order Control Barrier Function (CBF) as the core [16], this study designs a safety filter at the velocity level to achieve real-time collision risk avoidance and feasible control signal projection. When defining the robot configuration, let $p_i(q)$ be the end point of link *i*. To quantify the safety margin between the manipulator and obstacles, the minimum margin function is defined as:

$$c(q)=\min_{j}\min_{i\in\{0,\ldots,n-1\}}\left(\operatorname{dist}\left(o_j,\left[p_i(q),p_{i+1}(q)\right]\right)-\tilde{r}_j\right), \tag{24}$$

where $\operatorname{dist}\left(o,[a,b]\right)$ denotes the Euclidean distance from a point o to a line segment $[a,b]$, and $\tilde{r}_j$ is the preset minimum safety distance. Therefore, a geometric collision occurs between the manipulator and obstacles when $c(q)<0$; the manipulator is in the safe region when $c(q)\geq 0$; and a larger value of $c(q)$ indicates a higher safety margin.

Treating $c(q)$ as the safety function, we require that the time derivative of the safety function satisfies a non-negativity constraint. This ensures that the manipulator always moves in the direction of increasing safety margin when starting from a safe state, avoiding entry into collision areas:

$$\dot{c}(q)=\nabla c(q)^{\mathrm{T}}\dot{q}\geq-\alpha\left(c(q)\right), \tag{25}$$

where $\alpha(\cdot)$ is a class $\mathcal{K}$ function (a linear function $\alpha(c)=\rho c$, $\rho>0$ is adopted in this study). Its role is to constrain the decay rate of the safety function: when $c(q)$ approaches 0, $\dot{c}(q)\geq 0$ is forced to be positive, making the manipulator move away from collision areas rapidly.

The essence of the safety filter is a Quadratic Programming (QP) problem with linear inequality constraints [22]. The objective is to find a feasible joint velocity $\dot{q}_{\text{safe}}$ that is closest to the desired joint velocity $\dot{q}_{\text{des}}$ by MPPI, under the premise of satisfying the CBF constraints and the physical constraints of joint velocity. The constructed QP optimization problem is as follows:

$$\dot{q}_{\text{safe}}=\arg\min_{\dot{q}}\left\|\dot{q}-\dot{q}_{\text{des}}\right\|, \tag{26}$$

$$\text{s.t.}\quad \nabla c(q)^{\mathrm{T}}\dot{q}+\alpha\left(c(q)\right)\geq 0. \tag{27}$$

This QP problem has a closed-form solution and can be solved in real time using the Lagrangian method, thus satisfying the real-time requirement for online trajectory planning. When $\nabla c^{\mathrm{T}}\dot{q}_{\text{des}}\geq-\alpha(c)$, the constraint is not activated, and the solution is $\dot{q}_{\text{safe}}=\dot{q}_{\text{des}}$; when the constraint is activated, $\dot{q}_{\text{des}}$ is orthogonally projected onto the boundary of the constraint half-space to obtain the feasible solution:

$$\dot{q}_{\text{safe}}=\dot{q}_{\text{des}}-\frac{\alpha(c)+\nabla c(q)^{\mathrm{T}}\dot{q}_{\text{des}}}{\left\|\nabla c(q)\right\|_2^2+\delta}\nabla c(q), \tag{28}$$

 doi:10.20944/preprints202603.0086.v1

where $\delta > 0$ is a numerical stabilization term.

3.3.2. Algorithm Implementation

PG-MPPI integrates MPPI with a learned policy to enable safe and efficient planning. The overall implementation steps are presented in Table 1.

Table 1. PG-MPPI Algorithm.

Algorithm 1 PG-MPPI Pseudocode

Require: Dynamics $x_{t+1} = f(x_t, u_t)$; horizon H; samples K; temperature λ; noise covariance Σ; stage cost $\ell(x, u)$; terminal cost $\ell_f(x)$; learned policy $\pi_\phi(a \mid s)$; control constraint set $\mathcal{U}$; safety filter $\mathcal{F}_{\text{safe}}(\cdot)$.
Ensure: u_t applied to the robot at each time step.
Initialize nominal control sequence $U = \{u_0, \ldots, u_{H-1}\}$.
for $t = 0, 1, 2, \ldots, K$ **do**
  Observe current state x_t.
  Policy-guided nominal (warm start)
  Predict a nominal rollout under the learned policy:
    $\hat{x}_0 \leftarrow x_t$
  for $i = 0$ to $H - 1$ **do**
    $\hat{u}_i \leftarrow \mathbb{E}[\pi_\phi(\cdot \mid \hat{x}_i)]$
    $\hat{x}_{i+1} \leftarrow f(\hat{x}_i, \hat{u}_i)$
  end for
  Set the nominal sequence $U \leftarrow \{\hat{u}_0, \ldots, \hat{u}_{H-1}\}$.
  MPPI update
  $U \leftarrow \text{MPPI}(x_t, U; f, \ell, \ell_f, H, K, \lambda, \Sigma, \mathcal{U})$
  Safety filtering and execution
  $u_t^\star \leftarrow u_0$
  $u_t^{\text{safe}} \leftarrow \mathcal{F}_{\text{safe}}(x_t, u_t^\star)$
  Apply u_t^{safe} to the robot, obtain next state x_{t+1}.
  Receding horizon shift
  $U \leftarrow \{u_1, u_2, \ldots, u_{H-1}, u_{H-1}\}$
end for

In the offline phase, a policy $\pi_\phi(a \mid x)$ is trained in a simulation environment, where a CDF-based safety term guides the policy to learn obstacle avoidance and keep away from collision-prone regions. In the online phase, firstly, the CD-SAC policy is used to perform forward rollout to generate a nominal control sequence within the receding horizon H, which is then set as the initial nominal control sequence for MPPI. Next, following the standard MPPI update algorithm, the processes of sampling, rollout, cost calculation and weighted update are executed to obtain an optimized control sequence. The control signal at the initial time step of the optimized control sequence is extracted and fed into the safety filter $\mathcal{F}_{\text{safe}}(x_t, u_0)$, yielding a feasible control signal that satisfies all hard constraints. Finally, the feasible control signal is applied to the manipulator to drive the robot to complete the motion of the current time step.

Compared with traditional MPPI or RL methods, PG-MPPI integrates the advantages of both and features the following prominent characteristics:

Guided sampling and variance reduction. The nominal control of standard MPPI is usually based on the translation of the previous time step or simple heuristics, which is prone to falling into local minima or failing under complex constraints. PG-MPPI leverages the RL policy to provide a high-quality warm-start sequence, concentrating the sampling distribution near the globally optimal solutions. This greatly improves the sampling effectiveness and convergence speed;

Global planning capability and fast local correction. The RL policy learns a long-horizon value function through offline training, enabling it to handle tasks that require global information. In contrast, MPPI can perform high-frequency, real-time local correction on policy outputs, effectively responding to unencountered obstacles or model errors in offline training;

Complementary safety and constraint handling. RL training based on CDF enables the policy to internalize a soft safety preference, so that the output initial trajectory, even if not perfect, is always close to the safe region. On this basis, MPPI further guarantees the feasibility and safety of the

 doi:10.20944/preprints202603.0086.v1

trajectory at the execution level through hard constraint projection and collision cost penalty, forming a dual assurance mechanism of policy soft guidance + optimization hard constraint.

4. Experiments

To verify the effectiveness and superiority of the PG-MPPI algorithm, the SIASUN T12B six-joint manipulator is taken as the experimental object. A platform is built based on the MuJoCo [23] simulation framework, and environmental obstacles are modeled using CDF [19]. Multi-scenario comparative experiments are designed to conduct comparative analyses of PG-MPPI with SF-MPPI (the standard MPPI algorithm with a safety filter) and SF-SAC (the standard SAC algorithm with a safety filter). All experiments are conducted on a workstation equipped with an Intel i7-14650HX processor and an NVIDIA RTX 4050 graphics card, with the simulation time step set to 0.01s.

4.1. Offline Policy Learning

The environment is shown in Figure 3, where the motion target is a green sphere with coordinates (0.6, 0.2, 0.3). A cross-shaped obstacle centered at (0.8, 0.0, 0.5) and consisting of 13 red spheres with a radius of 0.05, is present in the workspace.

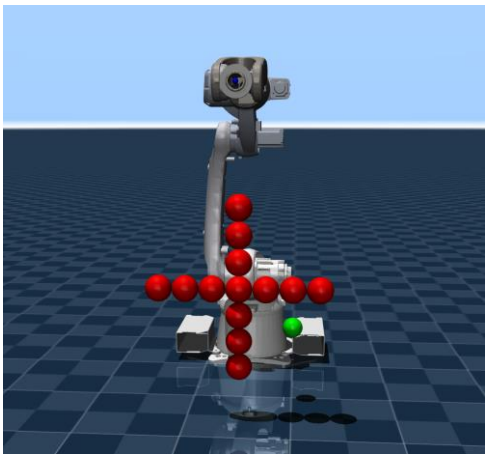

Figure 3. Manipulator and obstacle simulation.

Based on the simulation environment built in Section 4.1 and the designed constraint discounting mechanism, the proposed CD-SAC algorithm is adopted to conduct offline training for the actor network and critic network. The core parameters of the networks are shown in Table 2.

Table 2. CD-SAC Parameters.

Type	Parameters	Value	Function
Constraint Discount	velocity bound	1rad/s	Joint velocity constraint
	acceleration bound	2rad^2/s	Joint acceleration constraint
	$p^{\max}$	1	Termination probability calculation in Eq.(19)
	τ_c	0.99	Exponential moving average decay rate in Eq.(20)
SAC Network	hide dim	256	Number of neurons per layer for actor/critic MLP
	learning rate	3e-4	Learning rate of Adam optimizer
	γ	0.99	Baseline discount factor of SAC in Eq.(21)
	batch size	256	Batch size for each network update

 doi:10.20944/preprints202603.0086.v1

	action repeat	5	Number of physics simulation steps per RL step
Environment Reward	reach tolerance	0.03	position error threshold for task success
	max ep steps	2500	Maximum steps per training episode
	success bonus	10.0	Extra reward
	total steps	200000	Total environment interaction steps

The evolution curves of the core metrics during the training process are shown in Figure 4. In the early stage of training, since the policy had not yet explored the effective action space, the episode return remained at a low level with significant fluctuations. As the training iterations proceeded, the policy performance exhibited a rapid upward trend: the episode return rose sharply from a significantly negative value and tended to converge at approximately 50,000 steps, indicating that the policy had learned a control strategy to complete the end-effector positioning task at the minimum cost.

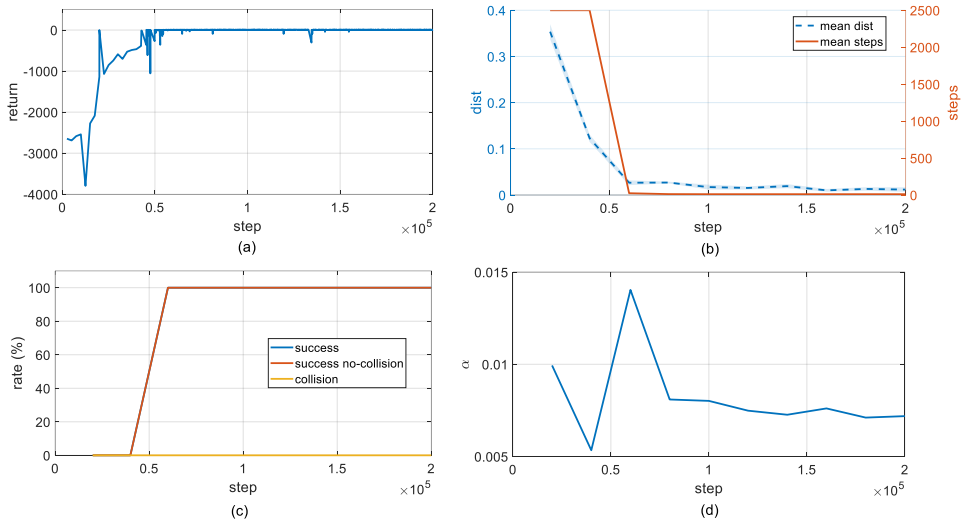

Figure 4. Metric Curves During Training. (**a**) Episode Return; (**b**) Mean Distance; (**c**) Success Rate; (**d**) Temperature Parameter.

From the perspective of quantitative evaluation metrics, the mean distance (average distance between the end-effector and the target) was approximately 0.35 m at the initial stage and then decreased exponentially; it stably decreased to less than 0.02 m after 50,000 steps, satisfying the task accuracy requirement at all times. Meanwhile, the mean steps (average execution steps per episode, right axis) plummeted from a value close to the maximum limit to a small value. This synchronous change clearly demonstrates that the policy behavior has efficiently transitioned from the timeout exploration/target unreachable mode in the early training stage to the fast positioning/early successful termination modes, verifying the fast convergence of the algorithm and the stability of its performance in the later stage.

In terms of safety, the collision rate remained nearly zero throughout the entire training cycle. Correspondingly, both the task success rate (success) and collision-free success rate (success no-collision) reached 100% synchronously after 50,000 steps and remained stable. This result fully indicates that, benefiting from the CDF-based safety guidance mechanism proposed in Section 3.2.1, the policy can not only stably complete the primary task but also autonomously avoid obstacles during motion, achieving a deep integration of task performance and safety.

In addition, the variation trend of the adaptive temperature parameter reflects the dynamic balance between exploration and exploitation of the policy: after a brief rebound in exploration intensity during the convergence stage, the temperature parameter gradually decreases and stabilizes

 doi:10.20944/preprints202603.0086.v1

at approximately 0.007. This shows that the algorithm actively reduces the exploration entropy in the late training stage and converges to the optimal deterministic control strategy, ensuring the robustness of the output trajectory.

In summary, the experimental results verify that the proposed CD-SAC algorithm can achieve a high task success rate and a near-zero collision rate with a small number of training iterations, and maintain excellent performance stability in the later training stage, which fully balances the efficiency, safety and control stability of manipulator motion.

4.2. Obstacle Avoidance Trajectory

Based on the converged CD-SAC policy network trained offline, this paper configures the core parameters of the PG-MPPI online planner, with the specific values presented in Table 3. In the real-time execution phase, the algorithm adopts a two-layer time-scale architecture: the optimal policy obtained offline serves as the global prior and is updated in a rolling with a period of 0.1 s; guided by this prior, MPPI performs sampling and local optimization at a high frequency of 0.01 s to generate the final control commands.

Table 3. PG-MPPI parameters.

Parameters	Value	Function
policy update	0.1s	Update cycle of the global prior policy
horizon	25	Length of the predictive receding horizon
samples number	200	Number of sampled trajectories for MPPI
λ	0.6	Temperature parameter
standard noise	0.6	Standard deviation of Gaussian noise

By independently implementing the SF-MPPI, SF-SAC and PG-MPPI trajectory planning algorithms, we obtain the motion trajectories of the robot end-effector, as shown in Figure 5, where the motion trajectories are marked with yellow curves.

 doi:10.20944/preprints202603.0086.v1

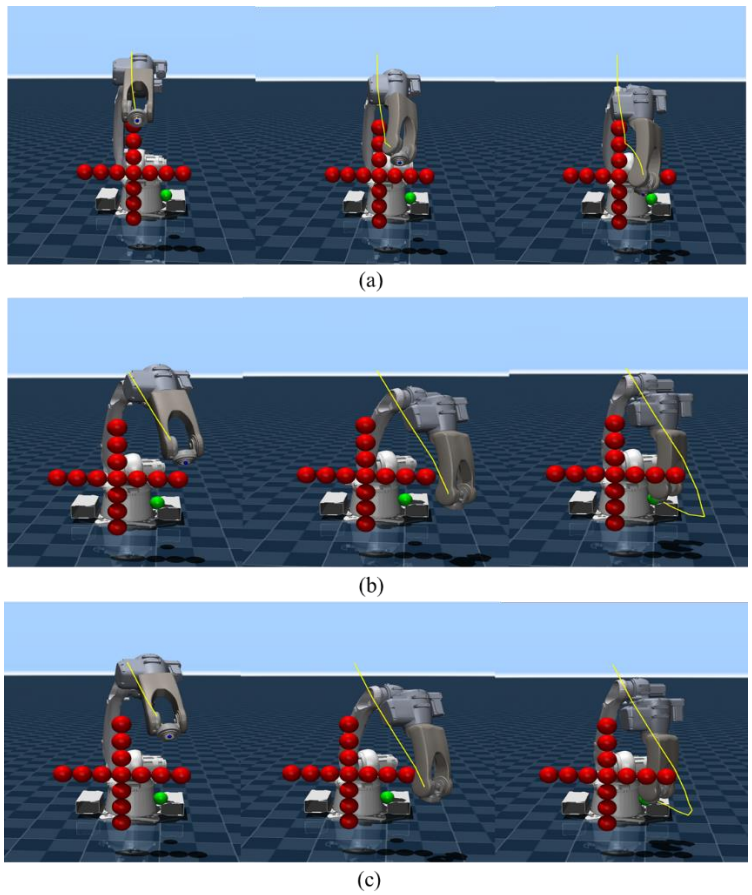

Figure 5. Motion Trajectories in Standard Obstacle Scenario. (**a**) SF-MPPI; (**b**) SF-SAC; (**c**) PG-MPPI.

SF-MPPI achieves hard-constraint collision avoidance for local obstacles via the safety filter, yet its myopic optimization property makes it prone to falling into local minima in the absence of long-horizon guidance. This is manifested by the trajectory terminating non-targeted in front of obstacles, thus failing to complete the end-effector positioning task. In contrast, both SF-SAC and PG-MPPI have internalized the prior of obstacle distribution in the training environment and are able to plan globally optimal obstacle avoidance paths. Experimental results show that these two algorithms can not only reach the target position accurately but also maintain a safe distance between all links of the manipulator and obstacles throughout the entire motion process.

The joint kinematic response curves of the PG-MPPI algorithm are shown in Figure 6. Throughout the entire motion cycle, the velocity and acceleration of all joints are strictly constrained within the preset physical boundaries with no constraint violations. Meanwhile, the velocity trajectories exhibit a continuous and smooth characteristic with low fluctuations, verifying that the proposed algorithm ensures physical feasibility while achieving excellent motion smoothness.

 doi:10.20944/preprints202603.0086.v1

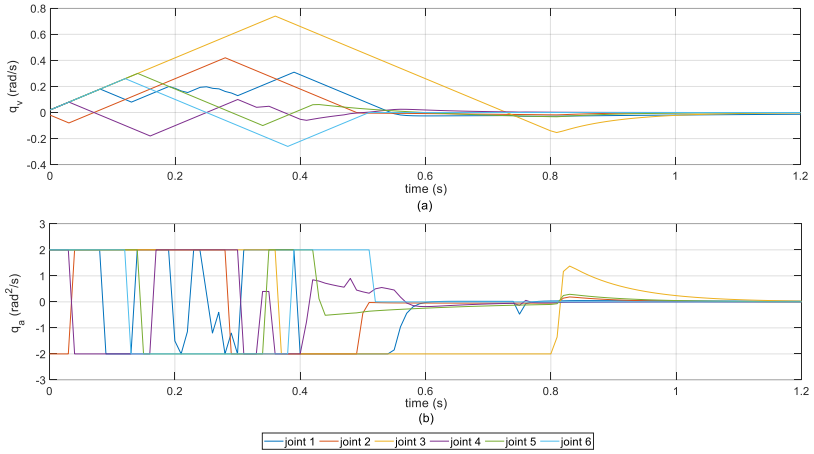

Figure 6. Joint Motion Curves of PG-MPPI. (**a**) Joint Velocity; (**b**) Joint Acceleration.

A simplified obstacle scenario is presented in Figure 7, where all three algorithms achieve collision-free target arrival. Due to the lack of global policy guidance, SF-MPPI can only complete path planning through local obstacle avoidance, resulting in a narrow safety margin between the trajectory and obstacles. In contrast, SF-SAC and PG-MPPI leverage the offline-trained global planning prior to generate obstacle avoidance trajectories with sufficient safety margins and reach the target position successfully.

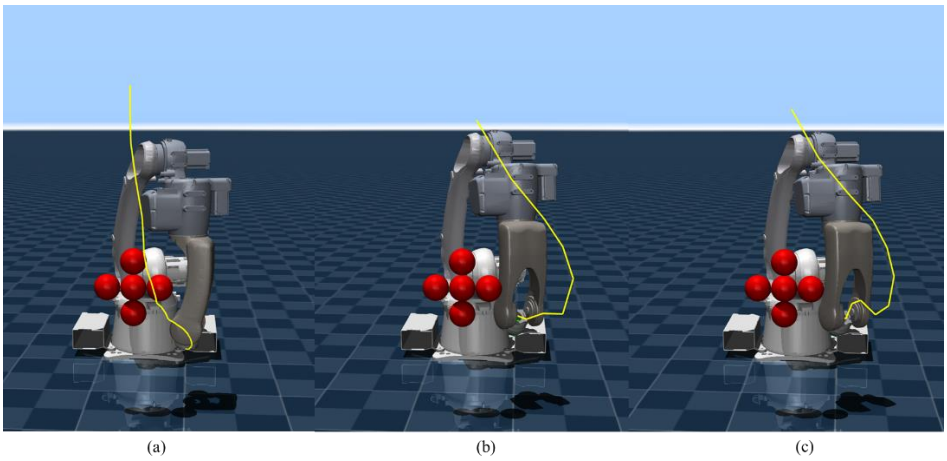

Figure 7. Motion Trajectories in Simplified Obstacle Scenario. (**a**) SF-MPPI; (**b**) SF-SAC; (**c**) PG-MPPI.

A reconfigured obstacle layout based on the standard environment is shown in Figure 8, where both SF-MPPI and SF-SAC expose their inherent limitations: limited by local optimization bias, SF-MPPI tends to get trapped in dead zones in obstacle-dense areas, leading to task failure; the offline policy of SF-SAC suffers from generalization failure due to the unseen obstacle distribution, resulting in trajectory collisions. On the contrary, through the collaborative mechanism of global policy guidance and local online optimization, PG-MPPI not only avoids potential collision traps using prior knowledge but also makes adaptive adjustments to unmodeled obstacles via real-time iteration of MPPI, thus generating a collision-free feasible trajectory and successfully completing the positioning task.

 doi:10.20944/preprints202603.0086.v1

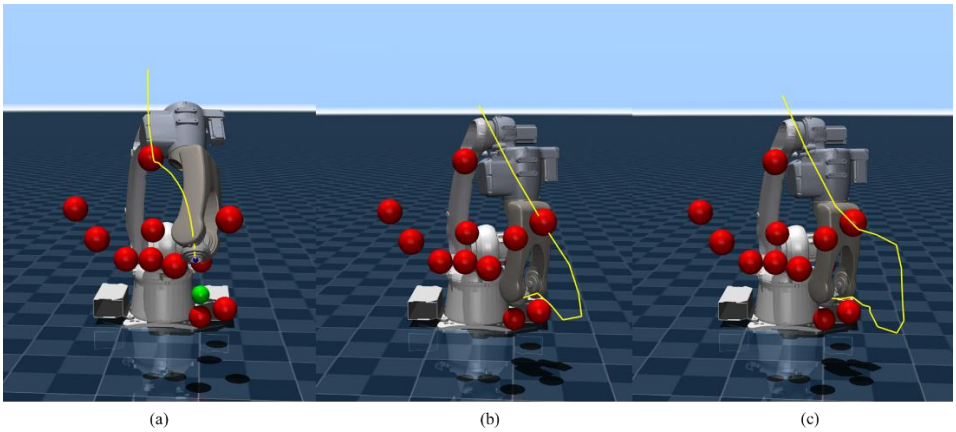

Figure 8. Motion Trajectories in Complex Obstacle Scenario. (**a**) SF-MPPI; (**b**) SF-SAC; (**c**) PG-MPPI.

To quantitatively evaluate the generalization and obstacle avoidance performance of the three algorithms, we randomly generate 10 typical initial joint configurations of the manipulator and test them in both the standard and complex obstacle simulation environments. We count the collision-free target arrival success rates of the three algorithms, with the results presented in Table 4.

Table 4. Success Rate of Typical Initial Configurations (%).

Method	SF-MPPI	SF-SAC	PG-MPPI
Normal Obstacle	20	100	100
Complex Obstacle	20	60	100

In the standard obstacle environment used for training, both SF-SAC and PG-MPPI achieve a 100% collision-free success rate with comparable performance. However, in the more challenging complex obstacle environment, PG-MPPI still maintains a 100% success rate by virtue of the collaborative mechanism of global policy guidance and local real-time optimization, outperforming the SF-SAC algorithm significantly. The above results fully demonstrate that the proposed algorithm has superior environmental generalization ability and dynamic obstacle avoidance performance.

Some representative trajectories planned by the PG-MPPI algorithm in the above experiments are presented in Figure 9. Multiple trajectories generated for different initial joint configurations can flexibly adapt to the obstacle distribution in the environment, generate collision-free paths with sufficient safety margins, and guide the manipulator to reach the target position accurately. This fully demonstrates the algorithm's robustness in trajectory planning and environmental adaptability under different initial conditions.

 doi:10.20944/preprints202603.0086.v1

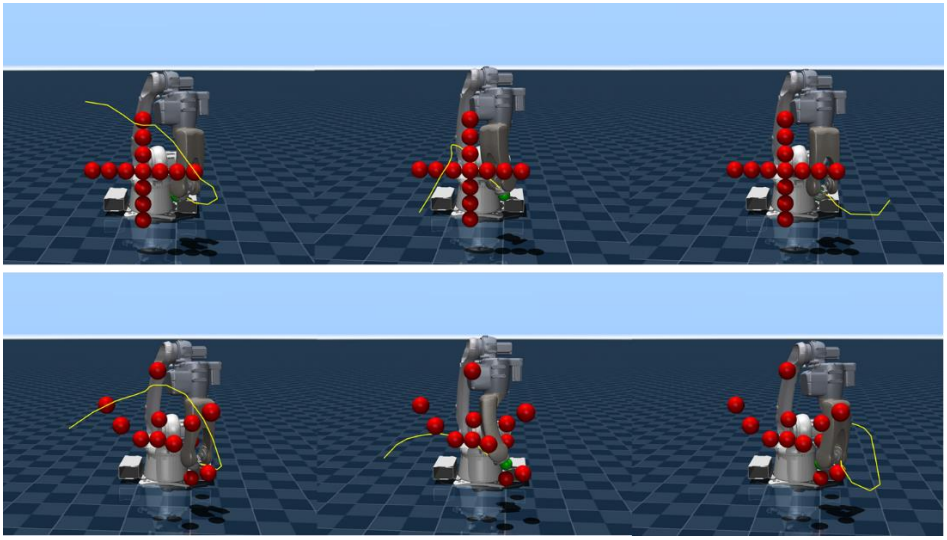

Figure 9. Motion Trajectories Based on PG-MPPI.

Based on the above qualitative and quantitative experimental analyses, the PG-MPPI algorithm proposed in this paper achieves global obstacle avoidance while completing dynamic local obstacle avoidance. Moreover, through the dual guarantee of the constraint discounting mechanism and safety filtering, it imposes hard constraints on the velocity and acceleration of joint trajectories, effectively ensuring the smoothness and stability of the manipulator's motion trajectory.

5. Conclusions

Aiming at the problems of manipulator motion planning in complex dynamic environments, this study proposes a PG-MPPI algorithm that integrates global policy guidance and local model predictive optimization. By incorporating a pre-trained CD-SAC policy network, the PG-MPPI algorithm provides a high-quality initial sampling distribution for the MPPI module, which effectively addresses the problems of low sampling efficiency and susceptibility to local optima for traditional MPPI in high-dimensional spaces. Experimental results show that in the standard obstacle environment, both the proposed algorithm and the baseline algorithms achieve a 100% collision-free target arrival success rate. In the more challenging complex obstacle environment, however, the PG-MPPI algorithm still maintains a 100% success rate, significantly outperforming the SF-SAC algorithm. This fully validates the algorithm’s excellent generalization ability and robust obstacle avoidance performance in unseen environments. Future work will further explore the deployment of this framework on physical hardware platforms and its integration with an environmental perception module to enable real-time adaptive planning for unknown dynamic obstacles.

6. Patents

Author Contributions: Conceptualization, L.L.; software, L.L.; validation, L.L.; writing—original draft preparation, L.L.; writing—review and editing, L.L.; supervision, X.W.; project administration, C.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Major Special Project of Innovation Consortium, grant number 2023JH1/11200014.

Data Availability Statement: The experimental data and source code associated with this study are available at the following repository: https://github.com/FantasyRobot/rl_mppi.

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

 doi:10.20944/preprints202603.0086.v1

References

1. Liu, J.; Yap, H.J.; Khairuddin, A.S.M. Review on motion planning of robotic manipulator in dynamic environments. *J. Sens.* **2024**, *1*, 5969512.
2. Koptev, M.; Figueroa, N.; Billard, A. Reactive collision-free motion generation in joint space via dynamical systems and sampling-based MPC. *Int. J. Robot. Res.* **2024**, *43*, 2049-2069.
3. Luo, S.; Zhang, M.; Zhuang, Y.; et al. A survey of path planning of industrial robots based on rapidly exploring random trees. *Front. Neurorobot.* **2023**, *17*, 1268447.
4. Schulman, J.; Ho, J.; Lee, A.X.; et al. Finding locally optimal, collision-free trajectories with sequential convex optimization. In Proceedings of the Robotics: Science and Systems 2013, Berlin, Germany, 24 June 2013.
5. Mayne, D.Q. Model predictive control: Recent developments and future promise. *Automatica* **2014**, *50*, 2967-2986.
6. Williams, G.; Drews, P.; Goldfain, B.; et al. Information-theoretic model predictive control: Theory and applications to autonomous driving. *IEEE Trans. Robot.* **2018**, *34*, 1603-1622.
7. Belvedere, T.; Ziegltrum, M.; Turrisi, G.; et al. Feedback-MPPI: fast sampling-based MPC via rollout differentiation–adios low-level controllers. *IEEE Robot. Autom. Lett.* **2025**, *11*, 1-8.
8. Qu, Y.; Chu, H.; Gao, S.; et al. RL-driven MPPI: Accelerating online control laws calculation with offline policy. *IEEE Trans. Intell. Veh.* **2023**, *9*, 3605-3616.
9. Ezeji, O.; Ziegltrum, M.; Turrisi, G.; et al. BC-MPPI: A Probabilistic Constraint Layer for Safe Model-Predictive Path-Integral Control. In Proceedings of Agents and Robots for Reliable Engineered Autonomy 2025, Bologna, Italy, 25 October 2025.
10. Tamizi, M.G.; Yaghoubi, M.; Najjaran, H. A review of recent trend in motion planning of industrial robots: MG Tamizi et al. *Int. J. Intell. Robot. Appl.* **2023**, *7*, 253-274.
11. Stan, L.; Nicolescu, A.F.; Pupăză, C. Reinforcement learning for assembly robots: A review. *Proc. Manuf. Syst.* **2020**, *15*, 135-146.
12. Romero, A.; Aljalbout, E.; Song, Y.; et al. Actor–Critic Model Predictive Control: Differentiable Optimization Meets Reinforcement Learning for Agile Flight. *IEEE Trans. Robot.* **2025**, *42*, 673-692.
13. Baltussen, T.; Orrico, C.A.; Katriniok, A.; et al. Value Function Approximation for Nonlinear MPC: Learning a Terminal Cost Function with a Descent Property. In Proceedings of the 2025 IEEE 64th Conference on Decision and Control, Rio De Janeiro, Brazil, 10 December 2025.
14. Hansen, N.; Wang, X.; Su, H. Temporal difference learning for model predictive control. *arXiv preprint* **2022**, 2203.04955.
15. Liu, P.; Zhang, Y.; Wang, H.; et al. Real-time collision detection between general SDFs. *Comput. Aided Geom. Des.* **2024**, *111*, 102305.
16. Ames, A.D.; Coogan, S.; Egerstedt, M.; et al. Control barrier functions: Theory and applications. In Proceedings of the 2019 18th European Control Conference, Naples, Italy, 15 June 2019.
17. Almubarak, H.; Sadegh, N.; Theodorou, E.A. Safety embedded control of nonlinear systems via barrier states. *IEEE Control Syst. Lett.* **2021**, *6*, 1328-1333.
18. Li, Y.; Miyazaki, T.; Kawashima, K. One-Step Model Predictive Path Integral for Manipulator Motion Planning Using Configuration Space Distance Fields. *arXiv preprint* **2025**, 2509.00836.
19. Li, Y.; Chi, X.; Razmjoo, A.; et al. Configuration space distance fields for manipulation planning. *arXiv preprint* **2024**, 2406.01137.
20. Crestaz, P.N.; De Matteis, L.; Chane-Sane, E.; et al. TD-CD-MPPI: Temporal-Difference Constraint-Discounted Model Predictive Path Integral Control. *IEEE Robot. Autom. Lett.* **2025**, *11*, 498-505.
21. Haarnoja, T.; Zhou, A.; Abbeel, P.; et al. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10 July 2018.
22. Ames, A.D.; Xu, X.; Grizzle, J.W.; et al. Control barrier function based quadratic programs for safety critical systems. *IEEE Trans. Autom. Control* **2016**, *62*, 3861-3876.

 doi:10.20944/preprints202603.0086.v1

23. Todorov, E. Convex and analytically-invertible dynamics with contacts and constraints: Theory and implementation in mujoco. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation, Hong Kong, China, 31 May 2014.