Control Barrier Function Augmentation in Sampling-based Control Algorithm for Sample Efficiency

Chuyuan Tao†, Hunmin Kim†, Hyungjin Yoon∗, Naira Hovakimyan†, and Petros Voulgaris∗

Abstract—For a nonlinear stochastic path planning problem, sampling-based algorithms generate thousands of random sample trajectories to find the optimal path while guaranteeing safety by Lagrangian penalty methods. However, the sampling-based algorithm can perform poorly in obstacle-rich environments because most samples might violate safety constraints, invalidating the corresponding samples. To improve the sample efficiency of sampling-based algorithms in cluttered environments, we propose an algorithm based on model predictive path integral control and control barrier function (CBF). The proposed algorithm needs fewer samples and time-steps and has a better performance in cluttered environments compared to the original model predictive path integral control algorithm.

I. INTRODUCTION

Path planning problem aims to find an efficient and collision-free path for a robot from a start position to a target position. There have been proposed various deterministic model-based approaches as solutions to this problem, such as model predictive control (MPC) [1], [2], and Bernstein polynomials [3]. However, these methods usually require calculating the derivative of the dynamics and the cost functions, which makes them computationally expensive when trying to solve nonlinear stochastic optimization problems. Some cost functions might not be differentiable over the entire state space, making them even more complicated.

Sampling-based algorithms use sample trajectories simulated from the dynamical models. The types of sampling-based algorithms include: the rapidly exploring trees (RRT) in [4] constructs the tree incrementally from samples drawn from the configuration space; the probabilistic roadmap in [5], [6] samples configuration in free configuration space and uses a local planner to connect them; the Markov chain Monte Carlo algorithm (MCMC) in [7] is used to approximate the posterior distribution of a parameter of interest by randomly sampling a probabilistic space; and the model predictive path integral control (MPPI) algorithm in [8] uses random diffusion trajectories to determine the stochastic optimal trajectory. These methods were shown to be effective in practical robotic applications [9], [10], where parallel computing is applicable to sample thousands of trajectories fast in parallel.

This work has been supported by the National Science Foundation (CNS-1932529) and UIUC STII-21-06.

†Chuyuan Tao, Hunmin Kim, and Naira Hovakimyan are with the Department of Mechanical Science and Engineering, University of Illinois at Urbana-Champaign, USA. {chuyuan2, hunmin, nhovakim}@illinois.edu

∗Hyungjin Yoon and Petros Voulgaris are with the Department of Mechanical Engineering, University of Nevada, Reno, USA. {hyungjiny, pvoulgaris}@unr.edu

However, the uniformly randomized sampling algorithms can become sample-inefficient in extreme circumstances [11], such as in a small region connecting two large free space areas, i.e., narrow passage problems. It takes more time for sampling-based algorithms to identify a safe path passing through the narrow passage, though the portion of the narrow passage is small compared to the whole configuration space. To address this issue, RRT was developed to use the previous exploration paths and generate incremental random paths starting from the previous terminal position [12]. Alternatively, an adaptive sampling method that adaptively adjusts the sample region to avoid collisions was proposed in [13] as an extension of RRT. In this paper, we aim to develop an MPPI-CBF algorithm that restricts the sampling distribution in the MPPI algorithm [14] by using the CBF to improve the sample efficiency of sampling-based algorithms for narrow passage problems. We trade off the computation time for solving a CBF-based optimization problem at every time step to guarantee sampling from a safe region. Thus, the proposed MPPI-CBF algorithm needs fewer samples and time steps to achieve a similar performance as compared to the MPPI algorithm.

Related Work. Stochastic model predictive control (SMPC) is an optimal control strategy for nonlinear stochastic systems subject to chance constraints. As one of the SMPC strategies, the MPPI algorithm provides an effective solution for finite horizon nonlinear stochastic optimization problems through parallel computing, which allows real-time control of a wide class of robotic applications [14]. The MPPI algorithm has been widely used in nonlinear path planning problems. In [15], the author uses machine learning algorithms to handle more general nonlinear control systems. Most existing MPPI algorithms use negative rewards as soft constraints to avoid collision. Another way to guarantee collision avoidance in path planning problems is to use CBF constraints [16]. The CBF methods can also be applied to systems with uncertainties. In [17], the author uses CBF to guide the reinforcement learning (RL) algorithm to explore a safe policy set. In [18], the author uses CBF to formulate a method that guarantees the safety of the system. Furthermore, in [19], a framework generating optimal control algorithms while guaranteeing safety on the multi-agent system is discussed.

II. PROBLEM STATEMENT

Consider the stochastic dynamic system to be in the following form:

\[ dx = f(x_t, t)dt + G(x_t, t)(dW(t) + u(x_t, t)dt), \]  (1)
where $x_t \in \mathbb{R}^n$ represents the state of an agent at time $t$, and let $u(x_t, t) \in \mathbb{R}^m$ be the control input of the system, and $dW(t) \in \mathbb{R}^p$ be a Brownian motion disturbance on the dynamical system.

We assume that the finite time horizon optimization problem has a quadratic control cost and an arbitrary state-dependent cost. The value function $V(x_t, t)$ is then defined as

$$
\min_u \mathbb{E}_Q \left[ \phi(x_T) + \int_t^T (q(x_t, t) + \frac{1}{2} u^T R(x_t, t) u) dt \right],
$$

(2)

where $R(x_t, t)$ is a positive definite matrix, $\phi(x_T)$ denotes a terminal cost, and $q(x_t, t)$ is a state-dependent cost. $\mathbb{E}_Q$ and $\mathbb{E}_P$ represent the expectations of the trajectories taken with the controlled dynamic system (1) and the uncontrolled dynamics of the system, respectively. We assume that the operating environment is obstacle rich and let $C$ represent a known safe set, i.e., if $x_t \in C$ for $\forall t \geq 0$, then the system is safe.

Problem 1: Given $x_0 \in C$, the problem is to develop a sampling-based algorithm for the optimization problem in (2) subject to (1), while guaranteeing $x_t \in C$ for $\forall t \geq 0$.

III. Control Barrier Function Based Model Predictive Path Integral Control (MPPI-CBF)

To address Problem 1 for cluttered environments, we develop a MPPI-CBF algorithm, where CBF is applied to determine the sampling distribution of each step in the MPPI algorithm to guarantee safety constraints in sampling based methods.

A. Model Predictive Path Integral Control (MPPI)

The MPPI algorithm [14], [15] is a sampling-based algorithm and provides a framework to solve nonlinear control problems efficiently for non-cluttered environments. By parallelizing the sampling step using Graphics Process Unit (GPU), the MPPI algorithm could sample thousands of trajectories of nonlinear complex dynamical systems in milliseconds [8]. The algorithm does not require finding derivatives of the dynamical system. Instead, sampled control inputs are propagated through the nonlinear dynamical system, and this allows to evaluate the cost for the corresponding sample trajectory.

Using the importance sampling weights and Monte-Carlo estimate [14], the control update law is given as:

$$
u^*(x_t, t) = u(x_t, t) + \mathbb{E}_Q \left[ \exp\left(-\frac{1}{\lambda} S(\tau)\right) \delta u \right] \mathbb{E}_Q \left[ \exp\left(-\frac{1}{\lambda} S(\tau)\right) \right],
$$

(3)

where $\lambda$ is the temperature parameter of Gibbs distribution, $\delta u$ is the random control input, and $\tau$ is the sample trajectory. The above equation (3) can be approximated as:

$$
u^*(x_t, t) \approx u(x_t, t) + \frac{\sum_{k=1}^K \exp\left(-\frac{1}{\lambda} S(\tau_{i,t})\right) \delta u_{i,t}}{\sum_{k=1}^K \exp\left(-\frac{1}{\lambda} S(\tau_{i,t})\right)}. \tag{4}
$$

It is common to use an indicator function as a penalty in the cost-to-go function $S(\tau)$ in solving path planning problems. However, in narrow passage problems, the majority of the random sample trajectories will violate the safe constraints, and have a large negative penalty. Thus, the sample efficiency will become low and the algorithm needs more samples to finish the narrow path problems. We use the control barrier function instead of adding penalties in the cost function to guarantee safety and increase sample efficiency.

B. CBF-Based Compensating Control with MPPI

We first introduce a straightforward shielded MPPI-CBF algorithm to illustrate the necessity of our framework. The concept is akin to shielded RL [20], which uses the CBF to compensate for the unsafe actions of the controller.

The CBF algorithm uses a Lyapunov-like argument to provide a sufficient condition for ensuring forward invariance of the safe set $C$. The following defines the CBF function.

Definition 1: [16] Let $C \subseteq \mathbb{R}^n$ be the super-level set of a continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$. Then $h$ is a control barrier function if there exists an extended class $K_\infty$ function $\alpha$ such that for the control system:

$$
\sup_{u \in U} [L_f h(x) + L_g h(x) u] \geq -\alpha(h(x)), \tag{5}
$$

for all $x \in D$, $L_f h : \mathbb{R}^n \to \mathbb{R}$ and $L_g h : \mathbb{R}^n \to \mathbb{R}$ are the Lie derivatives of $h$.

We assume a nominal feedback controller is defined as $\nu_{\text{nominal}} = k(x)$ and formulate a quadratic programming problem (QP) based on the CBF constraints to guarantee safety:

$$
u_{\text{CBF}} = \arg \min_u \frac{1}{2} \| u - k(x) \|^2_2 \text{ s.t. } L_f h(x) + L_g h(x) u \geq -\alpha(h(x)). \tag{6}
$$

The shielded MPPI-CBF algorithm uses the CBF as a safe filter for the control output of the MPPI controller, and it is shown in Fig. 1. The control update law becomes

$$
u_k(x_t) = u_{\text{CBF}}(x_t, u_{\text{MPPI}}(x_t)). \tag{7}
$$

The MPPI algorithm provides the nominal control input, and using the CBF-based optimization to find a minimum control intervention renders the safe set forward invariant.

Note that comparing to the CBF optimization in (6), the constraint of the stochastic system is not deterministic anymore. Therefore, we calculate the trust region of the
control input by a chance constraint to ensure the safety of
the robot. Therefore, the stochastic quadratic programming
at each time step is:
\[
\arg \min_u \| u - u_{\text{MPPI}} \|^2
\]
\[
\text{s.t. } \Pr(L_g h(x) u \geq -\alpha(h(x)) - L_f h(x)) \geq 1 - \delta.
\]
\[
(8)
\]
A critical issue for this shielded MPPI-CBF algorithm is that the CBF constraints override the control inputs of the MPPI algorithm without considering the sampling distribution. Thus, in the presence of safety challenges, the CBF constraints hinder the MPPI algorithm from exploring a feasible region. As a result, the CBF-based compensation controller with the MPPI algorithm prevents the robot from completing the mission. Therefore, we develop an MPPI-CBF algorithm based on sampling in safe regions instead of sampling uniformly and overriding the control input.

C. CBF-Based Trust Region Sample Control with MPPI

Instead of using CBF as a safe filter for the control input, we propose a new MPPI-CBF algorithm using CBF to determine the safe regions to sample. First, we define the initial control input samples from a Gaussian distribution
\[
u_0 \sim \mathcal{N}(\mu_0, \Sigma_0).
\]
Next, we assume the safe control input regions can be approximated by a Gaussian distribution
\[
u_{i,t} \sim \mathcal{N}(\mu_{i,t}, \Sigma_{i,t}).
\]
Note the safe control input region is a time-varying and state-dependent set.

Also, comparing to the QP optimization problem in (8), no nominal control is provided. Hence, we define a new convex cost function for the mean and variance of the sampling distributions:
\[
\mathcal{L}(\mu, \Sigma) = \|\mu - \mu_0\|_1 + \|\Sigma - \Sigma_0\|_p,
\]
where \(\mu, \Sigma\) are the variables to optimize and \(\|\cdot\|_p\) represents any matrix norm. We will use the above cost function for the CBF-based optimization to calculate the smallest change to the original control input distribution. We will find the safe mean and variance for the new sample trajectories at each time step. This structure is illustrated in Fig. 2.

Combining with the probabilistic CBF constraints defined in (8), we reformulate the optimization problem as
\[
\arg \min_{\mu_{i,t}, \Sigma_{i,t}} \|\mu_{i,t} - \mu_0\|_1 + \|\Sigma_{i,t} - \Sigma_0\|_p
\]
\[
\text{s.t. } \Pr(L_g h(x_{i,t})\nu_{i,t} \geq -\alpha(h(x_{i,t})) - L_f h(x_{i,t})) \geq 1 - \delta.
\]
\[
(10)
\]
Let \(A_{i,t} = L_g h(x_{i,t}), b_{i,t} = -\alpha(h(x_{i,t})) - L_f h(x_{i,t}).\) Then the left hand side of the control barrier function constraint can be considered as a new random variable \(A_{i,t}\nu_{i,t}\) having mean \(A_{i,t}\mu_{i,t}\) and variance \(A_{i,t}\Sigma_{i,t}A_{i,t}^T\). Using the chance constraint with the trust region probability \(1 - \delta\), the control barrier function constraint is simplified to
\[
A_{i,t}\mu_{i,t} - cA_{i,t}\Sigma_{i,t}A_{i,t}^T \geq b_{i,t},
\]
\[
(11)
\]
where \(c\) is the confidence interval corresponding to the probability \(1 - \delta\). To apply the CBF on the mean and variance, we also need to consider another constraint that the variance matrix of the control input random variable \(u_i^T\) must be positive semidefinite \(\Sigma \succeq 0\). Then, the optimization problem (10) is reformulated to the following convex optimization problem:
\[
\arg \min_{\mu_{i,t}, \Sigma_{i,t}} \|\mu_{i,t} - \mu_0\|_1 + \|\Sigma_{i,t} - \Sigma_0\|_p
\]
\[
\text{s.t. } A_{i,t}\mu_{i,t} - cA_{i,t}\Sigma_{i,t}A_{i,t}^T \geq b_{i,t} \Sigma_{i,t} \geq 0.
\]
\[
(12)
\]
This convex problem can be further transformed into a semidefinite programming problem (SDP), which can be solved more efficiently [21].

Lemma 1: The convex optimization problem (12) can be cast as a SDP problem.

Proof: A standard SDP problem is written as
\[
\min \ C \cdot X
\]
\[
\text{s.t. } A_i \cdot X = b_i \quad \text{for } i = 1, \ldots, n
\]
\[
X \succeq 0,
\]
\[
(13)
\]
where \(X \in S^n\), with \(S^n\) being the space of real symmetric \(n \times n\) matrices. The vector \(b \in \mathbb{R}^n\), and the matrices \(A_i \in S^n\) and \(C \in S^n\) are given parameters. While the standard SDP problem can also be cast in various ways, one of the most common formats is the inequality form given in [22]:
\[
\min \ c^T x
\]
\[
\text{s.t. } \sum_{i=1}^m x_i A_i \preceq B,
\]
\[
(14)
\]
where \(c\) is a vector and \(A_i, B \in S^n\) are matrices. The first part of the cost function in (12) is a vector norm for the mean difference. We rewrite it as \(\sum_{i=1}^m |\mu_i - \mu_0|\), and redefine the variable \(s_i = |\mu_i - \mu_0|\). Then the cost function becomes a linear combination of the variables \(s_i\). Using \(A(x) = A_0 + x_1 A_1 + \cdots + x_n A_n\) to represent the symmetric matrix, where \(A_i \in \mathbb{R}^{m \times m}, i = 0, \ldots, n\), the second part of the cost function becomes \(\min \|A(x)\|_p\). This can be further
reformed as follows with a new variable \( t > 0 \):
\[
\min t
\text{s.t.}
\begin{bmatrix}
 t I \\
 A(x) T \\
 A(x)
\end{bmatrix} \geq 0.
\]
(15)

Then, the cost functions of the first and second parts are linear and the constraints in (12), and those in (15) follow the form of SDP in (14). This completes the proof.

For any \( \Sigma_0 \), there exists matrix \( P_0 \) such that \( \Sigma_0 = P_0 P_0^T \). Furthermore, for any matrix \( P, \Sigma = PP^T \) becomes a symmetric definite matrix. Using this fact, we can simplify the constraint on the variance matrix, and the problem (12) becomes
\[
\arg \min_{\mu, P} \| \mu - \mu_0 \|_1 + \| P - P_0 \|_F
\text{s.t.}
\begin{bmatrix}
 I \\
 \sqrt{\Sigma} P \\
 \sqrt{\Sigma} P
\end{bmatrix} \geq 0.
\]
(16)

The solution of optimization (16) provides the safe mean \( \mu_s \) and the safe variance \( \Sigma_s = PP^T \) of the sample regions. Next, we will sample a new control input based on the new distribution and propagate through the dynamics (1) to get the state \( \tau_{i,t} \) of the new safe trajectories \( \tau_{i} \). It guarantees the sample trajectories are in the safe set \( C \) defined by the control barrier function with probability \( 1 - \delta \) for each time step. We will calculate the cost of each sample trajectory by using function \( S(\cdot) \) and using the following equation to calculate the weight of each trajectory:
\[
\omega_{i,t} = \exp\left(-\frac{1}{\lambda} S(\tau_{i,t})\right).
\]
(17)

We use the control update law:
\[
u^*_i(x, t_i) \approx u(x, t_i, t_i) + \sum_{k=1}^{K} \omega_{i,t} \delta \mu_{i,t} = \frac{\sum_{k=1}^{K} \omega_{i,t} \delta u_{i,t}}{\sum_{k=1}^{K} \omega_{i,t}}.
\]
(18)

The complete CBF-MPPI is summarized in Algorithm 1.

IV. EXPERIMENTS

In this section, we implement the algorithm on a nonlinear control-affine system. We design two different path planning simulation scenarios, single-obstacle avoidance and multi-obstacle narrow passage problems. We validate the effectiveness of the proposed methodology in the aforementioned two scenarios, from the perspective of safety guarantees and algorithm performance, by comparing with the MPPI-based algorithm.

A. Unicycle Dynamics

We consider a two-dimensional unicycle model in the simulation. In particular, the dynamical model of a unicycle is represented as:
\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{\theta}
\end{bmatrix} = \begin{bmatrix}
\cos \theta & 0 & v + \delta_v \\
\sin \theta & 0 & \omega + \delta_\omega
\end{bmatrix},
\]
(19)

where \( x, y \) are the coordinates, \( \theta \) is the angle, \( v \) is the linear velocity and \( \omega \) is the angular velocity of the unicycle model. \( \delta_v, \delta_\omega \sim \mathcal{N}(\mu_0, \Sigma_0) \) denote the random perturbation to the control input. The time step for the discrete time control update law (4) is \( \Delta t = 0.05s \).

B. Unicycle Single-Obstacle Avoidance Problem

In the first simulation, the path planning problem starts from position \((0, 0)\) and aims to reach the target position \( p_d \) at \((x_d, y_d)\). The unicycle needs to avoid a circular obstacle centered at \((O_x, O_y) = (2.2, 2.0)\), with the radius of \( r = 0.5 \). This control objective is enforced as a safety-critical constraint through the following CBF \( h(x, y) = (x - O_x)^2 + (y - O_y)^2 - O_r^2 \). The running cost function is designed in the form of:
\[
q(x) = 10\|p_d - p\|^2_2 + \|v_d - v\|^2_2
\]
(20)

where \( v_d \) is the desired velocity, and \( p(x, y) \) is the current position of the unicycle. The initial mean and variance matrix of the noise distribution are: \( \mu_0 = [0, 0], \Sigma_0 = I \). The target is set at \((4, 4)\) and the desired velocity is \( v_d = 2 \).

We first use 100, 200, and 500 samples and 20 time steps in the simulation. The probability of the confidence interval is \( 1 - \delta = 99.8\% \). As Fig. 3a demonstrates, the MPPI-CBF algorithm can always drive the unicycle to the target position via an optimal path with various sample sizes.

Algorithm 1 MPPI-CBF algorithm

\begin{algorithm}
\begin{algorithmic}
\While{task is not completed}
\For{i ← 0 to K - 1}
\For{t ← 1 to T}
\State Solving SDP in (16) to get \( \mu_s, \Sigma_s \);
\State Generate control variations \( \delta \mu_{i,t} \sim \mathcal{N}(\mu_s, \Sigma_s) \);
\State Simulate discrete dynamic to obtain \( x_{i,t+1} \);
\State Calculate cost function \( S(\tau_i) = q(x_i) \);
\EndFor
\State Calculate terminal cost \( S(\tau_i) = \phi(x_{i,T}) \);
\EndFor
\State \( \beta \leftarrow \min_k [S(\tau_i)] \);
\State Get resample weights \( \omega_{i,t} \) using (17);
\State Update control input using \( \omega_{i,t} \) and \( \delta \mu_{i,t} \) using (18);
\EndWhile
\end{algorithmic}
\end{algorithm}

We repeat the same simulation with the MPPI algorithm for comparison. We use a similar cost function form but introduce an indicator function term for penalization whenever the trajectory violates the safety set; then the revised cost function is in the form of \( q(x) = 10\|p_d - p\|^2_2 + \|v_d - v\|^2_2 + 10000 \times 1_{p \in \mathcal{C}} \). where \( \mathcal{C} \) is the complement to the safe set \( C \) over \( \mathbb{R}^2 \). 1 is the indicator function, and \( C = \{ (x, y) | (x - O_x)^2 + (y - O_y)^2 - O_r^2 \geq 0 \} \). The result of the MPPI algorithm is shown in Fig. 3b. In this single-obstacle avoidance scenario, it is observed that the MPPI algorithm can also guarantee safety by simply adding an indicator term in the cost function. However, comparing to
the MPPI-CBF algorithm, the trajectories from the MPPI algorithm are more aggressive, which can be explained as the MPPI algorithm samples uniformly in the configuration space, whereas the MPPI-CBF algorithm samples in a safe but more conservative space. However, the MPPI-CBF algorithm has a more conservative result in this considered scenario.

Beyond the conservativeness of the running trajectory, there is no significant difference between the proposed algorithm and the baseline MPPI algorithm from the perspective of cost-reduction profile, which is illustrated in Fig. 5a.

![MPPI-CBF algorithm for single-obstacle simulation](a) MPPI-CBF algorithm for single-obstacle simulation  
![MPPI algorithm for single-obstacle simulation](b) MPPI algorithm for single-obstacle simulation

Fig. 3: Single-obstacle collision avoidance scenario with sample numbers 100, 200 and 500.

![100 sample trajectories of MPPI-CBF algorithm at time t = 0.35s](a) 100 sample trajectories of MPPI-CBF algorithm at time t = 0.35s  
![100 sample trajectories of MPPI algorithm at time t = 0.35s](b) 100 sample trajectories of MPPI algorithm at time t = 0.35s

Fig. 4: The first figure shows that the MPPI algorithm samples uniformly in the configuration space, and the second figure shows that the MPPI-CBF algorithm samples in a restricted safe space (red: greater costs, blue: lower costs).

We also plot all the sample trajectories at time $t = 0.35s$ in Fig. 4. The black circle represents the obstacle, and the sample trajectories are colored based on the costs. From the plots, we can conclude that the MPPI algorithm samples uniformly on the entire configuration space, and some sample trajectories violate the safety constraints. However, for MPPI-CBF, which chooses samples from a safe set with high probability, and from Fig. 4, it is observed that almost all the samples are in the safe set. Though conservative, the proposed MPPI-CBF algorithm provides more safe samples than the MPPI algorithm. This advantage will be more obviously shown in the narrow passage problem in the next section.

![The cost of the MPPI and the MPPI-CBF algorithm in the single-obstacle avoidance scenario](a) The cost of the MPPI and the MPPI-CBF algorithm in the single-obstacle avoidance scenario  
![The cost of the MPPI and MPPI-CBF algorithm in the multi-obstacle narrow passage simulation](b) The cost of the MPPI and MPPI-CBF algorithm in the multi-obstacle narrow passage simulation

Fig. 5: The cost of different algorithms with different sample numbers.

### C. Multi-Obstacle Narrow Passage Problem

We consider the same dynamical model as discussed in Section IV-A and further add multiple circular obstacles to simulate a narrow passage problem. The $n$ circular obstacles are located at $(O_{x_i}, O_{y_i})$ with radius of $O_r(i = 1, \ldots, n)$, respectively. The distance between each circle is much smaller comparing to the radius. The safe set for the space is defined as: $C = \{(x, y) | (x - O_{x_i})^2 + (y - O_{y_i})^2 - O_r^2 \geq 0, i = 1, \ldots, n \}$. We use the same cost function (20) and similar control barrier functions $h_i(x, y) = (x - O_{x_i})^2 + (y - O_{y_i})^2 - O_r^2$ in the single-obstacle avoidance scenario. The initial mean and variance of the noise distribution is: $\mu_0 = [0, 0], \Sigma_0 = I$.

We also use the same target position and desired velocity. For the MPPI algorithm, we use 40 time steps to guarantee that the algorithm can reach the target position, and for the MPPI-CBF algorithm, we use 20 time steps to save the computation time in the simulation.

The running trajectories solving the narrow passage problem using MPPI and MPPI-CBF algorithms are shown in Fig. 6. The proposed MPPI-CBF algorithm admits safety guarantee and a shorter travel time to the target. However, with the same amount of samples, the MPPI algorithm cannot find the shortest path to the target because most sampled trajectories violate the safety constraints and get penalized. In the MPPI-CBF algorithm, however, almost all sample trajectories lie in the safe set computed in the SDP optimization step, and thus the samples are more efficient, which explains why the MPPI-CBF algorithm needs fewer samples and time steps to solve the problem. As shown in Fig. 5b, with the same sample number, the MPPI-CBF algorithm has a more rapid cost reduction than the MPPI algorithm.

We also plot the sample trajectories at time $t = 3.5s$. The results are shown in Fig. 7a and Fig. 7b. The black circles represent the obstacles, and the sample trajectories
Fig. 6: We use three circular obstacles to simulate a narrow passage problem, which is denoted by the black circles. The target is in blue. We plot the trajectories from the MPPI and the MPPI-CBF algorithms, with sample numbers of 50 and 100.

are in different colors according to the costs. It is observed that most sample trajectories of the MPPI algorithm violate the safe constraints. However, in the MPPI-CBF algorithm, almost all sample trajectories are in the safe region, which validates our claim on the sample efficiency of the proposed MPPI-CBF algorithm.

(a) MPPI sample trajectories at time $t = 0.35s$  
(b) MPPI-CBF sample trajectories at time $t = 0.35s$

Fig. 7: The first figure shows that the MPPI algorithm samples uniformly in the configuration space, and the second figure shows that the MPPI-CBF algorithm samples from a restricted safe space (red: greater costs, blue: lower costs).

V. CONCLUSION

In this paper, we propose an MPPI-CBF algorithm that uses the control barrier function to increase the safe efficiency and guarantee the safety of the sampling-based MPPI algorithm. We demonstrate that our algorithm can guarantee safety in cluttered environments, with improved sample efficiency.

REFERENCES

[1] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, “A survey of motion planning and control techniques for self-driving urban vehicles,” *IEEE Transactions on intelligent vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

[2] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat, “Predictive active steering control for autonomous vehicle systems,” *IEEE Transactions on control systems technology*, vol. 15, no. 3, pp. 566–580, 2007.

[3] R. Choe, J. Puig, V. Cichella, E. Xargay, and N. Hovakimyan, “Trajectory generation using spatial pythagorean hodograph b´ezier curves,” in *AIAA Guidance, Navigation, and Control Conference*, p. 0597, 2015.

[4] S. M. LaValle and J. J. Kuffner Jr, “Randomized kinodynamic planning,” *The international journal of robotics research*, vol. 20, no. 5, pp. 378–400, 2001.

[5] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[6] L. E. Kavraki, M. N. Kolountzakis, and J.-C. Latombe, “Analysis of probabilistic roadmaps for path planning,” *IEEE Transactions on Robotics and Automation*, vol. 14, no. 1, pp. 166–171, 1998.

[7] A. E. Gelfand and A. F. Smith, “Sampling-based approaches to calculating marginal densities,” *Journal of the American statistical association*, vol. 85, no. 410, pp. 398–409, 1990.

[8] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, “Aggressive driving with model predictive path integral control,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1453–1440, IEEE, 2016.

[9] K. Yang, S. Moon, S. Yoo, J. Kang, N. L. Doh, H. B. Kim, and S. Joo, “Spline-based rrt path planner for non-holonomic robots,” *Journal of Intelligent & Robotic Systems*, vol. 73, no. 1, pp. 763–782, 2014.

[10] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[11] W. Wang, L. Zuo, and X. Xu, “A learning-based multi-rrt approach for robot path planning in narrow passages,” *Journal of Intelligent & Robotic Systems*, vol. 90, no. 1, pp. 81–100, 2018.

[12] T.-Y. Li and Y.-C. Shie, “An incremental learning approach to motion planning with roadmap management,” in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 4, pp. 3411–3416, IEEE, 2002.

[13] L. Jaillet, A. Yershova, S. M. La Valle, and T. Siméon, “Adaptive tuning of the sampling domain for dynamic-domain rts;”, in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2851–2856, IEEE, 2005.

[14] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, “Information-theoretic model predictive control: Theory and applications to autonomous driving,” *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1603–1622, 2018.

[15] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou, “Information theoretic mpc for model-based reinforcement learning,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1714–1721, IEEE, 2017.

[16] I. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, “Control barrier functions: Theory and applications,” in *2019 18th European Control Conference (ECC)*, pp. 3420–3431, IEEE, 2019.

[17] R. Cheng, G. Orosz, R. M. Murray, and J. W. Burdick, “End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 3387–3395, 2019.

[18] P. Zhao, Y. Mao, C. Tao, N. Hovakimyan, and X. Wang, “Adaptive robust quadratic programs using control lyapunov and barrier functions,” in *2020 59th IEEE Conference on Decision and Control (CDC)*, pp. 3353–3358, IEEE, 2020.

[19] L. Song, N. Wan, A. Galihawat, C. Tao, N. Hovakimyan, and E. A. Theodorou, “Generalization of safe optimal control actions on networked multi-agent systems,” 2021.

[20] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin, “A general safety framework for learning-based control in uncertain robotic systems,” *IEEE Transactions on Automatic Control*, vol. 64, no. 7, pp. 2737–2752, 2018.

[21] R. Peng and K. Tangwongsan, “Faster and simpler width-independent parallel algorithms for positive semidefinite programming,” in *Proceedings of the twenty-fourth annual ACM symposium on Parallelism in algorithms and architectures*, pp. 101–108, 2012.

[22] H. Wolpert, R. Saigal, and L. Vandenberghe, *Handbook of semidefinite programming: theory, algorithms, and applications*, vol. 27. Springer Science & Business Media, 2012.