MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Chance-Constrained Information-Theoretic Stochastic Model Predictive Control with Safety Shielding

Yin, Ji; Tsiotras, Panagiotis; Berntorp, Karl

TR2024-179 December 18, 2024

Abstract

We introduce a nonlinear stochastic model pre- dictive control path integral (MPPI) method that considers chance constraints on system states. The proposed belief-space stochastic MPPI (BSS-MPPI) applies Monte-Carlo sampling to evaluate state distributions resulting from underlying systematic disturbances, and utilizes a Control Barrier Function (CBF) inspired heuristic in belief space to fulfill the specified chance constraints. Compared to several previous stochastic predictive control methods, our approach applies to general nonlinear dynamics without requiring the computationally expensive system linearization step. Moreover, the BSS-MPPI controller can solve optimization problems without limiting the form of the objective function and chance constraints and is parallelizable. Results on a realistic race-car simulation study show significant reductions in constraint violation compared to some of the prior MPPI approaches, while being comparable in computation times.

IEEE Annual Conference on Decision and Control (CDC) 2024

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Chance-Constrained Information-Theoretic Stochastic Model Predictive Control with Safety Shielding

Ji Yin[1], Panagiotis Tsiotras[1], Karl Berntorp*

Abstract— **We introduce a nonlinear stochastic model predictive control path integral (MPPI) method that considers chance constraints on system states. The proposed belief-space stochastic MPPI (BSS-MPPI) applies Monte-Carlo sampling to evaluate state distributions resulting from underlying systematic disturbances, and utilizes a Control Barrier Function (CBF) inspired heuristic in belief space to fulfill the specified chance constraints. Compared to several previous stochastic predictive control methods, our approach applies to general nonlinear dynamics without requiring the computationally expensive system linearization step. Moreover, the BSS-MPPI controller can solve optimization problems without limiting the form of the objective function and chance constraints and is parallelizable. Results on a realistic race-car simulation study show significant reductions in constraint violation compared to some of the prior MPPI approaches, while being comparable in computation times.**

I. Introduction

Over a span of 30 years, 37 robot-related accidents were reported, with 27 incidents resulting in a worker's death between 1984 and 2013. This data underscore the necessity of safety protocols to prevent workplace fatalities [1]. The evolution of robotic systems from performing static manipulation tasks into more collaborative and dynamic, further emphasizes the need for robust safety processes [2]. To address safety and increase robot reliability, it is crucial to consider uncertainties in robot planning and control. Uncertainties, such as dynamical disturbances, can lead to unpredictable robot behavior, which, in turn, might pose risks to human operators or other robots in the proximity. For instance, an autonomous vehicle might fail to respond adequately to unexpected road conditions, endangering the safety of its occupants and other road users.

We propose a novel control approach that considers chance constraints in belief space, using path integral and control barrier functions (CBFs) theories. Specifically, we design a heuristic in belief space inspired by *discrete-time* CBFs (DCBFs) to push the probability of collision lower than a user-specified threshold and integrate it with the Shield Model Predictive Path Integral (S-MPPI) [3] to achieve real-time planning. We call the resulting control method the belief-space stochastic MPPI (BSS-MPPI). We use Monte-Carlo sampling to estimate the state distributions and extract the first two moments into the state-dependent part of the objective function of the optimization problem. Then, we solve for a solution fulfilling chance constraints for safety. However, BSS-MPPI can also integrate with other, non-sampling based uncertainty-propagation schemes, such as nonlinear Kalman filters. Our approach

a) solves a nonlinear stochastic optimal control problem by forward simulation, hence avoiding the computationally-expensive explicit optimization and linearization steps, and can be parallelized on multi-threaded CPUs or GPUs;
b) can be applied to more general nonlinear stochastic systems. Previous works that consider chance constraints mostly assume linear/linearized dynamics, linear constraints, and some specific form of disturbances (e.g., Gaussian), which limit the range of applicability in robotic systems;
c) designs CBF-inspired heuristics that provide improved scalability at the cost of less formal guarantees and apply them to high-dimensional belief space.

Related Work: The papers [4], [5] formulate covariance steering as a convex optimization problem and plan trajectories for linearized vehicle models with additive Gaussian noise. The work [6] reformulates obstacle-avoidance chance constraints using the signed distance function and applies the sequential convex programming (SCP) algorithm, and [7] considers the chance-constrained trajectory optimization problem by integrating the constraints into the cost functions and uses dynamic programming to obtain a solution. The papers [8] and [9] propose stochastic CBFs (SCBF) with safety guarantees, but with an implicit assumption that the control inputs and the dynamical system must be unbounded.

Compared to many other nonlinear controllers, such as SCP, iLQR, and iLQG, MPPI implements the optimal control problem by forward simulation using the original nonlinear dynamics and hence avoids the linearizations involved in the explicit optimal control solvers. MPPI does not restrict the form of the objective function, which can be nonconvex and discrete. However, the base MPPI uses only deterministic dynamics. To improve the robustness of MPPI, several variants of the algorithm have been developed [10], [11]. In particular, [12], [13] introduce penalty costs for constraint violations due to dynamical and environmental uncertainties to the MPPI objective function, but they do not provide theoretical guarantee for probablistic constraint satisfaction. The work [14] also uses chance constraints, but it assumes linear systems and Gaussian disturbances.

[1]Ji Yin and Panagiotis Tsiotras are with D. Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, GA. This work has been partially supported by ONR award N00014-23-1-2353 and NSF award CPS-2211755. Ji Yin was an intern at MERL at the time of this research and this support was not applied to the work at MERL. {jyin81,tsiotras}@gatech.edu

* Corresponding author: Karl Berntorp, Mitsubishi Electric Research Labs (MERL), Cambridge, MA, USA. Email: karl.o.berntorp@ieee.org.

II. SHIELD-MPPI REVIEW

Consider a general nonlinear dynamical system,

$$x_{k+1} = f(x_k, u_k), \tag{1}$$

where $x_k \in \mathcal{D} \subseteq \mathbb{R}^{n_x}$ is the state, $u_k \in \mathbb{R}^{n_u} \sim \mathcal{N}(v_k, \Sigma_\epsilon)$ is the control input with the mean control v_k at time step $k = 0, \ldots, K-1$ and constant covariance Σ_ϵ. Let $\phi(x)$ denote the terminal state cost, $q(x)$ the step state cost, and λ the weight for control cost. Then, S-MPPI solves

$$\min_{\mathbf{v}} J(\mathbf{v}) = \mathbb{E}\left[\phi(x_K) + \sum_{k=0}^{K-1}\left(q(x_k) + \frac{\lambda}{2} v_k^\intercal \Sigma_\epsilon^{-1} v_k\right)\right], \tag{2}$$

subject to (1) with initial condition $x_0 = x(0)$, and the first-order DCBF safety condition,

$$\sup_{v \in \mathcal{U}} h(f(x_k, v_k)) - h(x_k) \geq -p(h(x_k)), \tag{3}$$

for $k = 0, \ldots, K-1$, where the continuous DCBF $h(\cdot) : \mathbb{R}^n \to \mathbb{R}$ defines a safe set in the state space,

$$\mathcal{S} \coloneqq \{x \in \mathcal{D} | h(x) \geq 0\}. \tag{4}$$

The class-κ function $p(\cdot)$ is strictly increasing and $p(0) = 0$. Fulfilling (3) ensures that the system stays within $\mathcal{S}$, and S-MPPI achieves this by minimizing the violation of (3) using path integral and gradient-based optimizations [3].

Compared to MPPI [15], S-MPPI typically requires an order of magnitude less trajectories to achieve equivalently satisfactory performance. Despite its attractive properties, S-MPPI uses only deterministic dynamics. To remedy this, we introduce a CBF-inspired heuristic to S-MPPI, such that the resulting BSS-MPPI plans with model uncertainties.

III. PROBLEM FORMULATION

In this paper, the objective is to solve

$$\min_{\mathbf{v}} J(\mathbf{v}), \text{ subject to,} \tag{5a}$$

$$x_{k+1} = f(x_k, u_k, w_k), \quad u_k \sim \mathcal{N}(v_t, \Sigma_\epsilon), \tag{5b}$$

$$\Pr(x_k \in \mathcal{F}) > 1 - P_{\text{fail}}, \text{ for } k = 0, \ldots, K-1, \tag{5c}$$

$$x_0 = x(0), \tag{5d}$$

where the noise w_k in (5b) can take arbitrary forms, $\mathcal{F}$ in (5c) is the feasible region, and (5c) denotes the set of chance constraints where the probability of violating the constraint is below a specified threshold value P_{fail}.

IV. BELIEF-SPACE MPPI

We review the MPPI control update law and develop the BSS-MPPI algorithm by integrating the chance constraint (5c) using a proposed belief-space heuristic in discrete time.

A. MPPI Control

The novel BSS-MPPI controller has an objective function in the same form as the original MPPI. Hence, we can use the MPPI control update law. The MPPI algorithm solves an optimization problem by sampling control inputs and forward simulating a large number of trajectories. Assuming that the algorithm has M trajectory samples with prediction horizon K, for the problem (2) subject to (1), the MPPI algorithm has the following control update law,

$$\mathbf{v}^+ = \sum_{m=1}^{M} \omega_m \mathbf{u}^m / \sum_{m=1}^{M} \omega_m, \tag{6}$$

where the m^{th} sample control sequence $\mathbf{u}^m = \{u_0^m, \ldots, u_{K-1}^m\}$, $u_t^m \sim \mathcal{N}(v_k, \Sigma_\epsilon)$, and where

$$\omega_m = \exp\left(-\frac{1}{\lambda}(S_m - \beta)\right). \tag{7}$$

Note that $\beta = \min_{m=1,\ldots,M} S_m$ in (7) is introduced to prevent numerical overflow, and S_m is the cost of the m^{th} sample trajectory, given by,

$$S_m = \phi(x_K^m) + \sum_{k=0}^{K-1} q(x_k^m) + \gamma(v_k^m)^\intercal \Sigma_\epsilon^{-1} u_k^m. \tag{8}$$

B. CBF-Inspired Heuristic Considering Chance Constraints

1) Safe heuristic in belief space: To fulfill the chance constraint (5c), we combine $\bar{x} = \mathbb{E}[x]$ and state covariance Σ to form a belief-space system,

$$\hat{z} = \begin{bmatrix} \bar{x} \\ \text{vec}(\Sigma) \end{bmatrix}, \tag{9}$$

where $\hat{z} \in \mathcal{Z} \subseteq \mathbb{R}^{n_z}$, $n_z = n_x(n_x + 1)$. We then choose a safe set $\mathcal{S} \subseteq \mathcal{Z}$, such that,

$$\hat{z} \in \mathcal{S} \iff \Pr(x \in \mathcal{F}) > 1 - P_{\text{fail}}, \; P_{\text{fail}} \in (0, 1], \tag{10}$$

using the following assumption on the feasible region:

Assumption 1 *The feasible region $\mathcal{F}$ is defined by the intersection of Z inequalities, such that,*

$$\mathcal{F} \coloneqq \bigcap_{i=1}^{Z} \{x : c_i(x) < 0\}, \; i = 1, \ldots, Z. \tag{11}$$

Using Assumption 1 and the Boole-Bonferroni inequality [16], any state x_k subject to the constraints,

$$\Pr(c_i(x) \geq 0) \leq p_i, \quad i = 1, \ldots, Z, \tag{12a}$$

$$\sum_{i=1}^{Z} p_i \leq P_{\text{fail}}. \tag{12b}$$

also satisfy the original constraint (5c). From (12), we can approximate each chance constraint by

$$c_i(\bar{x}_k) + \nu_i \sqrt{\eta_i^\intercal \Sigma_k \eta_i} \leq 0, \tag{13}$$

where $\Sigma_k \in \mathbb{R}^{n_x \times n_x}$ is the covariance of x_k, $\eta_i = \nabla_x c_i(\bar{x}_k)$, and ν_i is the *back-off coefficient* [17]. Hence, following from (4) and (13), we obtain the following heuristic,

$$h_i(\hat{z}) = -c_i(\bar{x}) - \nu_i \sqrt{\eta_i^\intercal \Sigma \eta_i}, \tag{14}$$

with a superlevel set $\mathcal{S}_i$ containing all states satisfying (13),

$$\mathcal{S}_i = \{\hat{z} \in \mathcal{Z} | h_i(\hat{z}) \geq 0\}. \tag{15}$$

Consequently, the safe heuristic for Problem (5) is

$$h(\hat{z}) = \min(h_1(\hat{z}), \ldots, h_Z(\hat{z})), \tag{16}$$

with a corresponding safe set $\mathcal{S} \subseteq \mathcal{Z}$ being the intersection of all superlevel sets $\mathcal{S}_i$ for $i = 1, \dots, Z$, given by,

$$S = \bigcap_{i=1}^{Z} S_i. \tag{17}$$

Remark 1 *The back-off coefficient value ν_i is computed to ensure the probability level p_i in the chance constraint. One option is to use the Cantelli-Chebyshev inequality, which holds regardless of the underlying probability distribution resulting in $\nu_i = \sqrt{\frac{1-p_i}{p_i}}$. However, it can lead to relatively conservative bounds [17]–[19]. Alternatively, for approximately normal-distributed state trajectories, we can set $\nu_i = \sqrt{2}\mathrm{erf}^{-1}(1-2p_i)$ where erf^{-1} is the inverse error function. The paper [20] discusses an alternative, sampling-based approach for constraint tightening.*

2) Safety condition: Assume the belief-space state follows the system $\hat{z}_{k+1} = f_z(\hat{z}_k, v_k)$. The safety condition for the heuristic (16) is given by,

$$h(\hat{z}_{k+1}) - h(\hat{z}_k) \geq -p(h(\hat{z}_k)), \tag{18}$$

where $p(\cdot)$ is a class-κ function. Satisfying (18) gives us two properties (see [3] for proofs).

Property IV.1 *Given an initial condition $\hat{z}_0 \in \mathcal{S}$ and a control sequence $\{v_k\}_{k=0}^{\infty}$ such that all $(\hat{z}_k, v_k)$ pairs satisfy* (18)*, then $\hat{z}_k \in \mathcal{S}$ for all $k \in \mathbb{Z}_{\geq 0}$.*

Property IV.2 *Let $\hat{z}_0 \in \mathcal{Z} \setminus \mathcal{S}$ and let a control sequence $\{v_k\}_{k=0}^{\infty}$ such that, for all $k \in \mathbb{Z}_{\geq 0}$, the pair (z_k, v_k) satisfies* (18)*. Then the state z_k converges to the safe set $\mathcal{S}$ asymptotically.*

Remark 2 *If there always exists a control v such that* (18) *is satisfied for all $\hat{z} \in \mathcal{Z}$, $h(\hat{z})$ becomes a DCBF. The HJ reachability analysis, which is commonly used to find verified CBFs, is computationally intractable for systems of high dimensions [21]. Hence, finding CBFs for belief-space systems is difficult in many cases. The proposed BSS-MPPI minimizes violation of* (18) *and achieves an average* 98.7% *satisfaction rate of* (18) *in our simulations, effectively keeping system states within the safe set and significantly reduces collision rates.*

C. Augmented System

Note that the MPPI objective function (2) and the corresponding control update law (6)-(8) share the same state-dependent step cost $q(\cdot)$, which can be of arbitrary form. To integrate the DCBF with chance constraints into the MPPI objective function and the control update law, we introduce a system in belief space that includes the mean and covariance of the system (5b), then augment the new system to combine two states in one, such that the safety condition (18) can be included in the step running cost $q(\cdot)$.

1) Belief-space system: We can separate the system state following (5b) into the mean and disturbed parts,

$$x_k = \bar{x}_k + \tilde{x}_k. \tag{19}$$

Given a sequence of sampled controls $\mathbf{u} = [u_0, \dots, u_{K-1}]$, the mean state $\bar{x}_k$ follows the nominal system,

$$\bar{x}_{k+1} = \bar{f}(\bar{x}_k, u_k) = \mathbb{E}[f(x_k, u_k, w_k)], \tag{20}$$

and the covariance propagation evolves according to

$$\Sigma_{x_{k+1}} = \mathrm{Cov}[f(x_k, u_k, w_k)] = f_\Sigma(\Sigma_{x_k}, u_k). \tag{21}$$

We can then describe (5b) using the belief-space system (9),

$$\hat{z}_{k+1} = \begin{bmatrix} \bar{x}_{k+1} \\ \mathrm{vec}(\Sigma_{x_{k+1}}) \end{bmatrix} = \begin{bmatrix} \bar{f}(\bar{x}_k, u_k) \\ \mathrm{vec}(f_\Sigma(\Sigma_{x_k}, u_k)) \end{bmatrix} \tag{22a}$$

$$= f_z(\hat{z}_k, u_k). \tag{22b}$$

In some cases, as demonstrated in our simulations, only part of the uncertainty covariance matrix is needed, thus a smaller state space can be used. For linear systems with Gaussian additive disturbances and for systems with nonlinearities on a specific form, $\bar{f}$ and f_Σ can be expressed analytically [22]. For nonlinear systems, data-driven approaches, such as neural nets and Gaussian processes [23]–[25] can be used to model the mean and covariance propagations. In this work, we propose to apply Monte-Carlo sampling to estimate the empirical mean and covariance propagation, such that,

$$\bar{x}_{k+1} = \mathbb{E}[x_{k+1}] \approx \frac{1}{N} \sum_{n=0}^{N-1} x_{k+1}^n, \tag{23}$$

and,

$$\Sigma_{x_{k+1}} = \mathbb{E}\left[\tilde{x}_{k+1}\tilde{x}_{k+1}^\intercal\right] \approx \frac{1}{N-1} \sum_{n=0}^{N-1} (x_{k+1}^n - \bar{x}_{k+1})(x_{k+1}^n - \bar{x}_{k+1})^\intercal, \tag{24}$$

where $x_{k+1}^n = f(x_k^n, u_k, w_k^n)$ is the state of the n^{th} trajectory sample following the control sequence $\mathbf{u}$ at time $k+1$.

2) Augmented belief-space system: Since the costs $\phi(x_K)$ and $q(x_k)$ in the objective function (2) are only dependent on the state at a single time step, it is difficult to integrate the safety condition (18) directly, which includes two consecutive system states. To this end, we introduce, for each $k = 1, \dots, K$, the augmented state $z_k = (z_k^{(1)}, z_k^{(2)}) = (\hat{z}_k, \hat{z}_{k-1}) \in \mathbb{R}^{2n_z}$ and the corresponding augmented belief-space system.

$$z_{k+1} = \begin{bmatrix} z_{k+1}^{(1)} \\ z_{k+1}^{(2)} \end{bmatrix} = \begin{bmatrix} f_z(z_k^{(1)}, u_k) \\ z_k^{(1)} \end{bmatrix} = \begin{bmatrix} \hat{z}_{k+1} \\ \hat{z}_k \end{bmatrix}. \tag{25}$$

D. Safety-aware Objective Function

We can apply the MPPI algorithm to the augmented system (25) with cost,

$$\min_{\mathbf{v}} J(\mathbf{v}) = \mathbb{E}\left[\phi_a(z_K) + \sum_{k=0}^{K-1} \left(q_a(z_k) + \frac{\lambda}{2} v_k^\intercal \Sigma_\epsilon^{-1} v_k\right)\right], \tag{26}$$

to yield a sequence of optimal controls. Note that the step running cost $q_a(\cdot)$ is dependent on two consecutive states of the belief-space system (22). Taking a linear class-κ function, it follows from (18) that,

$$h(\hat{z}_k) - (1-\beta)h(\hat{z}_{k-1}) \geq 0, \tag{27}$$

where $\beta \in (0,1)$, and $h(\cdot)$ is designed following (14) and (16). The resulting safe condition violation cost is

$$C_{\text{safe}}(z_k) = C \, \max\{-h(z_k^{(1)}) + (1-\beta)h(z_k^{(2)}), 0\}. \tag{28}$$

Hence, the state-dependent costs are

$$q_a(z_k) = q(z_k^{(1)}) + C_{\text{safe}}(z_k), \tag{29a}$$

$$\phi_a(z_K) = \phi(z_K^{(1)}) + C_{\text{safe}}(z_K). \tag{29b}$$

We set $z_0^{(2)} = \hat{z}_{-1} = \hat{z}_0$, such that $C_{\text{safe}}(z_0) = 0$. The optimal control sequence can be calculated by (6), (7), and (8), using (29) as the state-dependent running costs. The modified objective function (26) integrates the chance constraint (5c) by converting it into a cost minimization problem, which minimizes (5a) while penalizing any violations of the safety condition (18) to fulfill (5c).

V. Algorithm

Algorithm 1 gives the pseudo-code for BSS-MPPI. Line 2 obtaines the estimated robot state, lines 3-4 initialize states and trajectory costs, line 5-7 sample control sequences, and lines 8-9 propagate the mean states and covariances.

In [4], [26], [27], the mean and covariance dynamics are modeled analytically and can be used to directly compute the next states. If the system is complex and highly nonlinear, such that the mean and covariance dynamics are difficult to model, lines 8-9 can be implemented by Algorithm 2, which utilizes Monte-Carlo sampling to approximate mean and covariance propagation and is generally applicable.

The disturbance w_k in Algorithm 2 can be either from a fixed or a conditional (e.g., state-dependent) distribution. Lines 10-15 in Algorithm 1 use the augmented belief space system (25) to evaluate the simulated trajectory costs $\tilde{S}^m$, and compute the optimal controls using the control update law (6). Line 16 sends the safe control command to the actuators, and line 17 resets the nominal control sequence for the next control iteration.

Algorithm 1: Belief-space stochastic MPPI Algorithm

Given: Shield-MPPI costs $q(\cdot), \phi(\cdot)$, parameters γ, Σ_ϵ;
Input: Initial control sequence $\mathbf{v}$
1 **while** *task not complete* **do**
2 $\quad \bar{x}_0, \Sigma_{x_0} \leftarrow$ *GetStateEstimate*();
3 $\quad$ **for** $m \leftarrow 0$ ***to*** $M-1$ *in parallel* **do**
4 $\quad\quad \bar{x}_0^m \leftarrow \bar{x}_0, \quad z_0^m \leftarrow [\bar{x}_0^\intercal, \text{vec}(\Sigma_{x_0}), \bar{x}_0^\intercal, \text{vec}(\Sigma_{x_0})]^\intercal, \quad \tilde{S}^m \leftarrow 0$;
5 $\quad\quad$ Sample $\epsilon^{\mathbf{m}} \leftarrow \{\epsilon_{\mathbf{0}}^{\mathbf{m}}, \ldots, \epsilon_{\mathbf{K-1}}^{\mathbf{m}}\}$;
6 $\quad\quad$ **for** $k \leftarrow 0$ ***to*** $K-1$ **do**
7 $\quad\quad\quad u_k^m \leftarrow v_k + \epsilon_k^m$;
8 $\quad\quad\quad \bar{x}_{k+1}^m \leftarrow \bar{f}(\bar{x}_k^m, u_k^m)$;
9 $\quad\quad\quad \Sigma_{x_{k+1}}^m \leftarrow f_\Sigma(\Sigma_{x_k}^m, u_k)$;
10 $\quad\quad\quad z_{k+1}^m \leftarrow [(\bar{x}_{k+1}^m)^\intercal, \text{vec}(\Sigma_{x_{k+1}}^m), (\bar{x}_k^m)^\intercal, \text{vec}(\Sigma_{x_k}^m)]^\intercal$
11 $\quad\quad\quad \tilde{S}^m \leftarrow \tilde{S}^m + q(\bar{x}_k^m) + \gamma v_k^\intercal \Sigma_\epsilon^{-1} u_k^m + C_{\text{safe}}(z_k^m)$;
12 $\quad\quad$ **end**
13 $\quad\quad \tilde{S}^m \leftarrow \tilde{S}^m + \phi(\bar{x}_K^m) + C_{\text{safe}}(z_K^m)$;
14 $\quad$ **end**
15 $\quad \mathbf{v}^{\text{safe}} \leftarrow$ *OptimalControl*$(\{\tilde{S}^m\}_{m=0}^{M-1}, \{\mathbf{u}^m\}_{m=0}^{M-1})$;
16 $\quad$ *ExecuteCommand*(v_0^{safe});
17 $\quad \mathbf{v} \leftarrow \mathbf{v}^{\text{safe}}$;
18 **end**

Algorithm 2: Mean and Covariance Propagation

Given: System with disturbance $f(x, u, w)$, noise distribution $p(w|x)$.
Input: $\bar{x}_k, \Sigma_{x_k}, u_k$;
for $n \leftarrow 0$ ***to*** $N-1$ *in parallel* **do**
$\quad$ Sample $x_k^n \sim \mathcal{N}(\bar{x}_k, \Sigma_{x_k}), w_k \sim p(w_k|x_k)$;
$\quad x_{k+1}^n = f(x_k^n, u_k, w_k)$
end
$\bar{x}_{k+1} \leftarrow \frac{1}{N}\sum_{n=0}^{N-1} x_{k+1}^n$
$\Sigma_{x_{k+1}} \leftarrow \frac{1}{N-1}\sum_{n=0}^{N-1}(x_{k+1}^n - \bar{x}_{k+1})(x_{k+1}^n - \bar{x}_{k+1})^\intercal$

VI. Simulation Study

We evaluate the proposed BSS-MPPI on a simulated car racing example, where the objective is to conclude a lap of a race course subject to minimizing a control objective. We develop the CBF-inspired heuristic for the application, and carry out a Monte-Carlo simulation study to evaluate the performance of BSS-MPPI (BSS-MPPI). We compare it with other MPPI-based control approaches, in particular, the original MPPI (MPPI, [15]) and S-MPPI (S-MPPI, [3]).

A. Experimental Setup

We use the AutoRally racing platform [28] to evaluate our method. The AutoRally is a 1m long, 0.4m wide electric vehicle with mass 22kg whose dynamics mimic a real vehicle. We model the dynamics using the discrete-time system (1), based on the planar single-track vehicle model [29]. The system state is $x = [v_X, v_Y, \dot{\psi}, \omega_F, \omega_R, e_\psi, e_Y, s]^\top$, where v_X is the longitudinal vehicle velocity, v_Y is the lateral vehicle velocity, $\dot{\psi}$ is the yaw rate, ω_F, ω_R is the front and rear wheel-speed, respectively, e_Y is the lateral deviation from the centerline, e_ψ is the yaw-angle deviation with respect to the centerline heading, and s is the path coordinate in the road-aligned frame. The control input is $u = [\delta, T]^\top$ where δ is the steering angle at the front wheel and T is the throttle. We use the Pacejka tire model and the friction ellipse to model combined slip [30].

We use the state-dependent running cost

$$q(x_k^m) = (x_k^m - x_g)^\top Q(x_k^m - x_g) + \mathbf{1}(x_k^m), \tag{30}$$

where $Q = \text{diag}(q_{v_X}, q_{v_Y}, q_{\dot{\psi}}, q_{\omega_F}, q_{\omega_R}, q_{e_\psi}, e_y, q_s)$ is the cost matrix, $x_g = \mathrm{diag}(v_g, 0, \ldots, 0)$, and

$$\mathbf{1}(x_k^m) = \begin{cases} 0, & \text{if } x_k^m \text{ satisfies the constraints,} \\ C_{\text{obs}}, & \text{otherwise,} \end{cases} \tag{31}$$

is the collision cost. Since BSS-MPPI extends S-MPPI by introducing chance constraints into the problem formulation, we mostly focused our evaluation on the ability of BSS-MPPI to satisfy the constraints under uncertain dynamics. We have executed 200 Monte-Carlo simulations where the vehicle is tasked to complete a lap of the AutoRally racetrack subject to zero-mean Gaussian process noise, $w_k \sim \mathcal{N}(0, \Sigma_w)$ with the reference velocity $v_g = 6\text{m/s}$.

B. Belief-Space Control Barrier Function Design

Assuming that the racing track has constant width $2w_{\mathrm{T}}$, we want to keep the vehicle's lateral deviation e_y from the track centerline bounded by $|e_y| \leq w_{\mathrm{T}}$ for some given collision probability δ. Hence, we can write the chance constraints as

$$P(e_y < w_{\mathrm{T}}) > 1 - \epsilon, \quad P(e_y > -w_{\mathrm{T}}) > 1 - \epsilon. \tag{32}$$

The chance constraints (32) are equivalent to

$$P(e_y \geq w_{\mathrm{T}}) \leq \epsilon, \quad P(e_y \leq -w_{\mathrm{T}}) \leq \epsilon. \tag{33}$$

It follows from (12a), (13) that (33) can be converted to deterministic chance constraints,

$$\bar{e}_y - w_T + \nu\sigma_y \leq 0, \quad \bar{e}_y + w_T - \nu\sigma_y \geq 0, \tag{34}$$

where σ_y is the standard deviation of e_y, obtained from the Monte-Carlo covariance propagation (see Algorithm 2). Since (34) is symmetric, we can form a single inequality, $|\bar{e}_y| \leq w_T - \nu\sigma_y$. Hence, we formulate the DCBF $h(x) = (w_T - \nu\sigma_y)^2 - e_y^2$, and the corresponding safe set is (4). We implement the chance constraints with the back-off coefficient $\nu = \sqrt{2}\mathrm{erf}^{-1}(1 - 2\epsilon)$ (see Remark 1), which was a good compromise between conservativeness and safety.

C. Simulation Results

Fig. 1 shows the trajectories for a set of runs produced by S-MPPI and BSS-MPPI when trying to manuever a lap of the race track. The trajectories produced by S-MPPI have a tendency to go close to the track boundaries and on several occasions also cross the track boundaries, indicating a subsequent crash. In contrast, owing to the chance constraints, BSS-MPPI trajectories tend to push closer to the track center, thus yielding safer trajectories.

We have run the different controllers using different tuning parameters, and for each set of parameters, we have executed the scenario in Fig. 1 for 100 Monte-Carlo runs using different noise realizations and initial conditions. Table I shows the crash and collision ratios for different number of trajectories for MPPI, S-MPPI, and BSS-MPPI, respectively. We define a collision to occur when the vehicle deviates from the middle of the lane with more than 1.8m (which defines the chance constraint for the lateral deviation), and similarly we define a crash to occur when the vehicle goes outside of the (assumed constant) lanewidth $w_{\mathrm{T}} = 2\text{m}$ from the center of the lane, indicating a severe constraint violation.

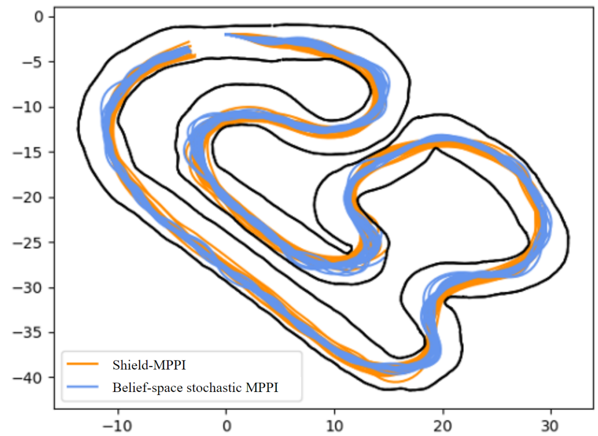

Fig. 1: S-MPPI and BSS-MPPI trajectory visualization for a set of Monte-Carlo runs where the objective is to conclude a lap clock-wise. The trajectories generated by BSS-MPPI consider disturbances and are more conservative, resulting in lower speeds but fewer collisions

TABLE I: Crash and collision ratio, and computational speed for 100 Monte-Carlo runs for a given cost function with $q_{e_y} = 0.1$, and a control horizon of 20 steps. When a crash occurs, the particular Monte-Carlo run is terminated. The simulations use an Nvidia RTX2060 GPU and the computational speed is the average time it takes to execute the respective method one time step.

Method	Crashes	Collisions	Speed [Hz]
MPPI, $M = 30,000$	93%	230%	58.1
S-MPPI, $M = 5,000$	8%	13%	55.3
S-MPPI, $M = 20,000$	8%	13%	42.9
BSS-MPPI, $MN = 100 \cdot 50$	4%	10%	43.1
BSS-MPPI, $MN = 500 \cdot 40$	0%	1%	39.7
BSS-MPPI, $MN = 1,000 \cdot 20$	0%	2%	39.1

With this notion, we can quantify the number of constraint violations, in addition to quantifying the number of times the controllers heavily violates the constraints.

Irrespective of the number of trajectories used, BSS-MPPI experiences substantially fewer crashes and collisions throughout the 100 Monte-Carlo runs. The vanilla MPPI (MPPI) crashes at almost all of the 100 Monte-Carlo runs and has on average 2.3 constraint violations per lap. The number of crashes and collisions for S-MPPI is more or less constant irrespective of the number of trajectories, while the safety of BSS-MPPI improves as the number of trajectories involved to approximately solve the optimal control problem increases. With a sufficiently large MN, the collisions and crashes almost diminish. Even if the number of trajectories used to approximate the covariance propagation N is large, BSS-MPPI is not able to recover appropriately if the number of sampled control sequences M is small.

Fig. 2 displays heatmaps of the crash and collision rates to traverse a lap of the track using BSS-MPPI with different number of sampled control sequences M and uncertainty evaluation trajectories N. It is clear that increasing N leads to fewer crashes, as the belief space trajectories and chance constraints are better approximated. However, only increasing N is not sufficient, since using a small M implies limited exploration of the optimal controls. Hence, the optimal choice is a trade-off between control exploration, uncertainty propagation, and computational resources.

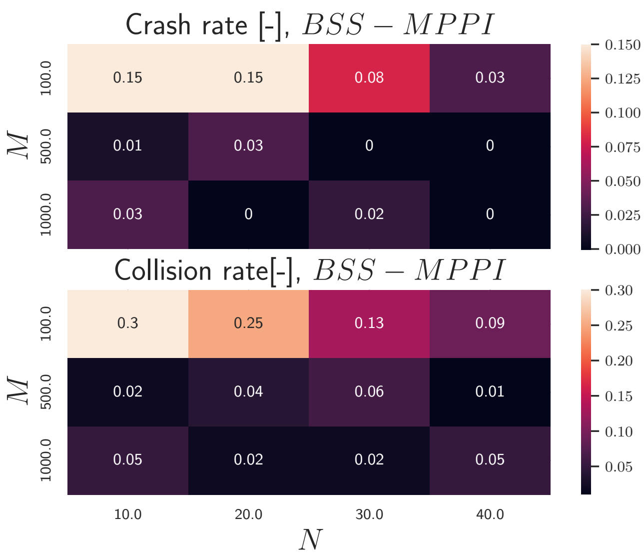

Fig. 2: Crash and collision rates as a function of the number of control sequences M and covariance-propagation trajectories N, with $q_{e_y} = 0.1$.

VII. Conclusion

We presented BSS-MPPI, which accounts for system uncertainty by leveraging a CBF-inspired heuristic to satisfy chance constraints. The method can handle nonlinear dynamics and solves the underlying nonlinear stochastic optimal control problem by forward simulation of control and uncertainty trajectories, thus avoiding explicit linearization and optimization steps. The results indicate that our method effectively reduces the number of constraint violations, while at the same time achieving computational times comparable to previous MPPI approaches.

References

[1] T. M. Anandan. Robot safety, everything but routine. [Online]. Available: https://www.automate.org/industry-insights/robot-safety-everything-but-routine#:~:text=Over%20a%2030,Bureau%20of%20Labor%20statistics

[2] G. Burdick. A closer look at the robotics safety process. [Online]. Available: https://ehsdailyadvisor.blr.com/2022/02/the-robotics-safety-process/#:~:text=Updated%3A%20Feb%2017%2C%202022%20Industrial,collaborative%20and%20mobile%20in%20nature

[3] J. Yin, C. Dawson, C. Fan, and P. Tsiotras, "Shield model predictive path integral: A computationally efficient robust mpc method using control barrier functions," *IEEE Robotics and Automation Letters*, vol. 8, no. 11, pp. 7106–7113, 2023.

[4] K. Okamoto and P. Tsiotras, "Optimal stochastic vehicle path planning using covariance steering," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2276–2281, 2019.

[5] J. Yin, Z. Zhang, E. Theodorou, and P. Tsiotras, "Trajectory distribution control for model predictive path integral control using covariance steering," in *International Conference on Robotics and Automation (ICRA)*, Philadelphia, PA, May 2022, pp. 1478–1484.

[6] T. Lew, R. Bonalli, and M. Pavone, "Chance-constrained sequential convex programming for robust trajectory optimization," in *2020 European Control Conference (ECC)*, 2020, pp. 1871–1878.

[7] M. Ono, M. Pavone, Y. Kuwata, and J. Balaram, "Chance-constrained dynamic programming with application to risk-aware robotic space exploration," *Autonomous Robots*, vol. 39, 12 2015.

[8] A. Clark, "Control barrier functions for stochastic systems," *Automatica*, vol. 130, p. 109688, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109821002089

[9] O. So, A. Clark, and C. Fan, "Almost-sure safety guarantees of stochastic zero-control barrier functions do not hold," 2023.

[10] H. Lee, T. Kim, J. Mun, and W. Lee, "Learning terrain-aware kinodynamic model for autonomous off-road rally driving with model predictive path integral control," *arXiv preprint arXiv:2305.00676*, 2023.

[11] K.-P. Kim and C.-H. Lee, "Adaptive weight model predictive path integral control for multiconstrained missile guidance," *Journal of Aerospace Information Systems*, pp. 1–18, 2024.

[12] J. Yin, Z. Zhang, and P. Tsiotras, "Risk-aware model predictive path integral control using conditional value-at-risk," 2022. [Online]. Available: https://arxiv.org/abs/2209.12842

[13] E. Arruda, M. J. Mathew, M. Kopicki, M. Mistry, M. Azad, and J. L. Wyatt, "Uncertainty averse pushing with model predictive path integral control," in *IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, Birmingham, UK, Nov 2017, pp. 497–502.

[14] I. M. Balci, E. Bakolas, B. Vlahov, and E. A. Theodorou, "Constrained covariance steering based tube-mppi," in *2022 American Control Conference (ACC)*, 2022, pp. 4197–4202.

[15] 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.

[16] A. Prékopa, "Boole-bonferroni inequalities and linear programming," *Operations Research*, vol. 36, no. 1, pp. 145–162, 1988.

[17] R. Quirynen and K. Berntorp, "Uncertainty propagation by linear regression kalman filters for stochastic nmpc," *IFAC-PapersOnLine*, vol. 54, no. 6, pp. 76–82, 2021, 7th IFAC Conference on Nonlinear Model Predictive Control NMPC 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896321013021

[18] D. Telen, M. Vallerio, L. Cabianca, B. Houska, J. V. Impe, and F. Logist, "Approximate robust optimization of nonlinear systems under parametric uncertainty and process noise," *J. Proc. Control*, vol. 33, pp. 140–154, 2015.

[19] M. M. Sean Vaskov, Rien Quirynen and K. Berntorp, "Friction-adaptive stochastic nonlinear model predictive control for autonomous vehicles," *Vehicle System Dynamics*, vol. 62, no. 2, pp. 347–371, 2024.

[20] A. P. Vinod and S. Di Cairano, "Sample quantile-based programming for non-convex separable chance constraints," in *2023 American Control Conference (ACC)*, 2023, pp. 1517–1522.

[21] I. M. Mitchell, "The flexible, extensible and efficient toolbox of level set methods," *J. Scientific Computing*, vol. 35, pp. 300–329, 2008.

[22] M. F. Huber, F. Beutler, and U. D. Hanebeck, "Semi-analytic Gaussian assumed density filter," in *Amer. Control Conf.* San Francisco, CA, Jun. 2011.

[23] 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)*, 2017, pp. 1714–1721.

[24] L. Hewing, J. Kabzan, and M. N. Zeilinger, "Cautious model predictive control using gaussian process regression," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 6, pp. 2736–2743, 2020.

[25] G. Cao, E. M.-K. Lai, and F. Alam, "Gaussian process model predictive control of an unmanned quadrotor," *Journal of Intelligent & Robotic Systems*, vol. 88, pp. 147–162, 2017.

[26] M. Goldshtein and P. Tsiotras, "Finite-horizon covariance control of linear time-varying systems," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 3606–3611.

[27] A. Jain, T. Nghiem, M. Morari, and R. Mangharam, "Learning and control using gaussian processes," in *2018 ACM/IEEE 9th Int. Conf. on Cyber-Physical Systems (ICCPS)*, 2018, pp. 140–149.

[28] B. Goldfain, P. Drews, C. You, M. Barulic, O. Velev, P. Tsiotras, and J. M. Rehg, "AutoRally: An open platform for aggressive autonomous driving," *IEEE Control Syst. Mag.*, vol. 39, no. 1, pp. 26–55, 2019.

[29] K. Berntorp, B. Olofsson, K. Lundahl, and L. Nielsen, "Models and methodology for optimal trajectory generation in safety-critical road–vehicle manoeuvres," *Veh. Syst. Dyn.*, vol. 52, no. 10, pp. 1304–1332, 2014.

[30] H. B. Pacejka, *Tire and Vehicle Dynamics*, 2nd ed. Oxford, United Kingdom: Butterworth-Heinemann, 2006.