Probably Approximately Correct Nonlinear Model Predictive Control (PAC-NMPC)
Adam Polevoy, Marin Kobilarov, and Joseph Moore, Member, IEEE

Abstract—Approaches for stochastic nonlinear model predictive control (SNMPC) typically make restrictive assumptions about the system dynamics and rely on approximations to characterize the evolution of the underlying uncertainty distributions. For this reason, they are often unable to capture more complex distributions (e.g., non-Gaussian or multi-modal) and cannot provide accurate guarantees of performance. In this letter, we present a sampling-based SNMPC approach that leverages recently derived sample complexity bounds to certify the performance of a feedback policy without making assumptions about the system dynamics or underlying uncertainty distributions. By parallelizing our approach, we are able to demonstrate real-time receding-horizon SNMPC with statistical safety guarantees in simulation and on hardware using a 1/10th scale rally car and a 24-inch wingspan fixed-wing unmanned aerial vehicle (UAV).

Index Terms—Integrated planning and control, planning under uncertainty, robot safety.

I. INTRODUCTION

NONLINEAR model predictive control (NMPC) has proven to be a powerful approach for controlling high-dimensional, complex robotic systems (e.g., [1], [2], [3], [4]). Nevertheless, although these methods can handle large state spaces, nonlinear dynamics, and system constraints, their performance can be adversely affected by the presence of uncertainty even in the context of real-time replanning [5]. A number of approaches have been proposed to compensate for this marginal robustness, the simplest of which is to generate a feedback policy to track the current receding-horizon plan (e.g., [2]). These approaches, however, do not account for uncertainty or closed-loop performance during the planning process. To address this, approaches such as robust NMPC (RNMPC) and stochastic NMPC (SNMPC) attempt to reason about the response of the policy to disturbances during planning. This is usually accomplished by making simplifying assumptions about the dynamics and disturbances. For instance, in RNMPC, disturbance sets may be approximated by ellipsoids [6], or robustness guarantees may require the existence of a control contraction metric (CCM) [7]; in SNMPC, uncertainty distributions are often approximated via Gaussians [8] or sampling [9].

In this letter, we present a sampling-based SNMPC algorithm capable of controlling stochastic dynamical systems without applying limiting assumptions to the structure of the stochastic dynamics or underlying uncertainty distributions. In addition, our approach leverages recently derived sample complexity bounds [10] to provide a probabilistic guarantee on system performance. Our algorithm builds on Probably Approximately Correct Robust Policy Search (PROPS) [11] to directly optimize an upper confidence bound on the expected cost and the probability of constraint violation for a receding-horizon feedback policy. Not only does this approach encourage robust policy generation, but the minimized bounds themselves provide a statistical guarantee for each planning interval. To achieve real-time performance, we use a graphics processing unit (GPU) to parallelize our algorithm. We then evaluate our approach in simulation and on hardware using a 1/10th scale rally car and a 24-inch wingspan fixed-wing UAV (Fig. 1).

Our contributions are:
1) A novel algorithm, PAC-NMPC, for receding horizon SNMPC with probabilistic performance guarantees.
2) A real-time implementation via GPU acceleration.
3) Demonstration of our algorithm on complex underactuated systems via simulation and hardware experiments.

II. RELATED WORK

Prior research addresses the impact of uncertainty on NMPC via robust NMPC (RNMPC) or stochastic NMPC (SNMPC). RNMPC represents uncertainty in the system as bounded disturbances. While min-max tube NMPC methods [6], [12], [13] and multi-stage NMPC approaches [14] have been proposed, constraint-tightening based tube NMPC [15] has proved to be the most common and computationally tractable RNMPC approach. Most of these approaches rely on tube size derived from a computed Lipschitz constant [16], [17], CCMs [7], [18], [19], or bounds on incremental stability [20]. These methods for computing invariant sets are often complex, may place significant restrictions on the dynamics, and can lead to overly conservative results. Recent work has proposed sampling-based reachability analysis for arbitrary continuously differentiable dynamics with less conservative results [21], [22].

Stochastic NMPC represents uncertainty as a probability distribution and optimizes the expectation over costs and constraint violations. Oftentimes, they optimize a feedback policy, rather than a control sequence. Our method, PAC-NMPC, falls into this category of approaches. Early approaches focused on a local iterative solution to the stochastic Hamilton-Jacobi-Bellman equation [23], [24], which relies on a first order approximation of the dynamics for noise propagation. Stochastic differential dynamic programming approaches [25] would later allow for second order approximation of the dynamics. More recent methods utilize the unscented transform to propagate noise through nonlinear dynamics [8], [26]. In [27], the authors leverage path integral control to generate open-loop trajectories and create a sampling-based NMPC algorithm. More recent extensions [9], [28] include a feedback policy for robust performance, [29] provides guarantees on free-energy growth, and [30] penalizes Conditional Value-at-Risk (CVaR).

Sampling-based stochastic optimal control methods are closely related to policy search in reinforcement learning. A general background and survey of common policy search methods can be found in [31]. In [11], researchers developed an approach for robust policy search using a Probably Approximately Correct (PAC) learning framework called PAC Robust Policy Search (PROPS) based on recently derived sample complexity bounds in [10]. Researchers have also recently explored (PAC)-Bayes theory to generate collision-avoidance policies with performance guarantees in novel environments [32] and achieve vision-based planning with motion primitives [33]. Our PAC-NMPC method builds on the work in [10] and [11] to develop a highly generalizable sampling-based stochastic feedback motion-planning algorithm that can provide statistical performance guarantees for a large class of uncertain dynamical systems.

III. BACKGROUND

Before presenting our approach for Probably Approximately Correct NMPC, we first review the recently derived PAC bounds for Iterative Stochastic Policy Optimization which are the foundation for our approach.

### A. Iterative Stochastic Policy Optimization

Consider the stochastic dynamics given by \( p(x_{t+1}|x_t, u_t) \) defined by a vector of state values \( x_t \in \mathbb{R}^{N_x} \), a vector of control inputs \( u_t \in \mathbb{R}^{N_u} \), and probability density \( p \). Iterative Stochastic Policy Optimization (ISPO) [10] formulates the search for a control policy \( u_t = \pi_t(x_t, \xi) \) as a stochastic optimization problem, where \( \xi \) is a set of policy parameters. Specifically, ISPO introduces a surrogate distribution \( p(\xi|\nu) \) where policy parameters \( \xi \) are dependent on distribution hyper-parameters \( \nu \). This induces a joint distribution

\[
p(\tau, \xi|\nu) = p(\tau|\xi) p(\xi|\nu)
\]

where \( p(\tau|\xi) \) represents the natural stochasticity of the system and is given as \( p(\tau|\xi) = p(x_0) \prod_{t=0}^{T} p(x_{t+1}|x_t, u_t), \) where \( u_t = \pi_t(x_t, \xi) \). Here \( \tau \) represents the discrete time trajectory sequence \( \{x_0, u_0, x_1, u_1, \ldots, u_{N_T}, x_{N_T+1}\} \), where \( N_T \) is the number of timesteps.

In general, the objective of ISPO is to solve the optimization problem \( \nu^* = \arg \min_{\nu} \mathcal{J}(\nu) \) where \( \mathcal{J}(\nu) = \mathbb{E}_{\tau, \xi \sim p(\cdot|\nu)}[\mathcal{J}(\tau)] \). To do this, ISPO iteratively samples from (1) and solves the optimization problem \( \hat{\nu}^*_{i+1} = \arg \min_{\nu} \tilde{\mathcal{J}}(\nu) + \alpha D(\nu, \hat{\nu}^*_i) \) for each iteration \( i \) until convergence. Here \( \tilde{\mathcal{J}}(\nu) \) is an empirical approximation of the expected cost, \( D \) is a distance between distributions, often the \( K\mathcal{L} \)-divergence, and \( \alpha > 0 \) is a hand-tuned weight or Lagrange multiplier [34] computed for a constraint on \( D(\nu, \hat{\nu}^*_i) \). Algorithm 1 is a general framework for ISPO.

### B. PAC Bounds for Stochastic Policy Search

Instead of directly minimizing an empirical approximation of the expected cost, it can be beneficial to minimize an upper confidence bound on the expected cost. Not only does such an approach encourage robust policies, but it also can provide guarantees on future performance. In this letter, we directly optimize the PAC bounds derived in [10]. While these bounds are derived in [10] and discussed in [11], we review them here since they are fundamental to our approach.

In [10] the presented PAC bound \( \mathcal{J}_t^+(\nu) \) takes the form

\[
\mathcal{J}_t^+(\nu) \triangleq \tilde{\mathcal{J}}_t(\nu) + \alpha d(\nu) + \Phi_\delta(\delta)
\]

where \( \tilde{\mathcal{J}}_t(\nu) \) is a robust estimator of the expected cost, \( d(\nu) \) is a distance between distributions, and \( \Phi_\delta(\delta) \) is a concentration-of-measure term which accounts for discrepancies between the true mean, \( \mathcal{J}(\nu) \), and the robust estimator, \( \tilde{\mathcal{J}}_t(\nu) \). For a particular choice of \( \tilde{\mathcal{J}}(\nu), d(\nu), \) and \( \Phi_\delta(\delta), \) [10] proves that \( \mathcal{J}_t^+(\nu) \) bounds the expected cost \( \mathcal{J}(\nu) \) with a probability of \( 1 - \delta \).

### Algorithm 1: Episodic ISPO.

1. Inputs: \( \nu_0^*, i \leftarrow 0; \)
2. while termination condition not met do
   3. for \( j = 1, \ldots, M \) do
      4. Sample Trajectory: \( (\tau_j, \xi_j) \sim p(\cdot, \cdot|\nu^*_i) \);
      5. Evaluate Cost: \( J_j = J(\tau_j) \);
      6. \( \hat{\nu}^*_{i+1} = \arg \min_{\nu} \tilde{\mathcal{J}}(\nu) + \alpha D(\nu, \hat{\nu}^*_i) \);
      7. \( i \leftarrow i + 1 \)
8. return \( \hat{\nu}^*_i \)
To derive an expression for $\tilde{J}_\alpha(\nu)$, we must estimate

$$J(\nu) = \mathbb{E}_{\tau, \xi \sim p(\cdot, \nu)} \left[ J(\tau) \frac{p(\xi | \nu)}{p(\xi | \nu_0)} \right]. \quad (3)$$

Because the likelihood ratio $\frac{p(\xi | \nu)}{p(\xi | \nu_0)}$ can be unbounded, a robust estimation technique is needed. Following [35], the expected value of a random variable $X$ can be approximated as $\mathbb{E}[X] \approx \frac{1}{\alpha M} \sum_{i=0}^{M} \psi(\alpha X_i)$ for some $\alpha > 0$, where $X_i$ is a sampled value of $X$ and $\psi(x) = \log(1 + x^2)$. Given $L$ prior surrogate distributions with hyper-parameters $\nu_0, \nu_1, \ldots, \nu_{L-1}$ and $M$ iid samples from each joint distribution, $(\tau_{0,0}, \xi_0), (\tau_{1,1}), \ldots, (\tau_{t-M}, \xi_{M}) \sim p(\cdot, \cdot | \nu)$, the robust estimate of the expected cost can then be written as

$$\tilde{J}_\alpha(\nu) \triangleq \frac{1}{\alpha LM} \sum_{i=0}^{M} \psi(\alpha \ell_{ij}) \quad (4)$$

$$\ell_{ij} = J(\tau_{ij}) \frac{p(\xi_{ij} | \nu)}{p(\xi_{ij} | \nu_i)} \quad (5)$$

To compute the distance between the distributions, the Renyi divergence, $D_2$, is used. $d(\nu)$ is given as

$$d(\nu) \triangleq \frac{1}{2} \sum_{i=0}^{L-1} \sum_{i=0}^{M} b^2 e^{D_2(p(\cdot | \nu))/D_2(p(\cdot | \nu_i))} \quad (6)$$

$$0 \leq J(\tau_{ij}) \leq b_i \quad \forall j = 0, \ldots, M \quad (7)$$

The concentration-of-measure term $\Phi_\alpha(\delta)$ is given as

$$\Phi_\alpha(\delta) \triangleq \frac{1}{\alpha LM} \log \frac{1}{\delta} \quad (8)$$

It tightens the bound as the number of samples increases and as the bound confidence, $1 - \delta$, decreases.

In addition to cost-functions, we can also incorporate the probability of constraint violation, $C(\nu)$, of the form $C(\nu) = \mathbb{P}(g(\tau) > 0) = \mathbb{E}_{\tau, \xi \sim p(\cdot, \nu)}[\mathbb{I}[g(\tau) > 0]]$, where $I$ is the indicator function. Similarly, we can derive $C_+^\alpha(\nu)$ which upper bounds $C(\nu)$ with probability $1 - \delta$.

To combine costs and constraints, we optimize a weighted sum of $J_\alpha(\nu)$ and $C_+^\alpha(\nu)$, where $\gamma > 0$ is a heuristically selected weighting coefficient. We solve

$$\nu_{t+1} = \nu^* = \arg \min_{\nu} \min_{\alpha > 0} \left( J_\alpha(\nu) + \gamma C_+^\alpha(\nu) \right) \quad (9)$$

using a GPU implementation of L-BFGS-B [36]. When optimizing the bounds, we used self-normalized importance weights to compute $\ell_{ij}$ in (5) and analytical gradients, as described in [11].

IV. APPROACH

To formulate a receding-horizon NMPC algorithm that leverages the PAC bounds in [10], we first address the stochastic trajectory optimization problem. We then extend this approach to enable real-time feedback motion planning.

A. PAC Stochastic Trajectory Optimization

To formulate stochastic trajectory optimization using PAC bounds, we consider a parameterization of a discrete-time open-loop policy $u_t = \pi_t(\xi) = \xi_t$, where $\xi = p(\xi_0, \ldots, \xi_{M})$.

Algorithm 2: PAC Stochastic Trajectory Optimization.

1. Inputs: $\delta_0$, $x_0$, $i \leftarrow 0$;

2. while termination condition not met do

   3. for $j = 1, \ldots, M$ do // GPU Parallelized

      4. $\tau_{ij}, \xi_j \sim p(\cdot, \cdot | \nu_0)$;

      5. $J_{ij} = J(\tau_{ij}), c_{ij} = c(\tau_{ij})$;

      6. $\nu_{t+1} \leftarrow \arg \min_{\nu_0} \min_{\alpha > 0} (J_\alpha(\nu) + \gamma C_+^\alpha(\nu))$;

      7. $i \leftarrow i + 1$;

   8. return $\nu^*, \tilde{J}_\alpha(\nu^*)$;

Algorithm 3: $\tau \sim p(\cdot | \xi, x_0)$.

1. Inputs: $\xi = [u_0^T, \ldots, u_{N_T}^T]^T, x_0$;

2. for $t = 0, \ldots, N_T$ do // Stochastic Rollout

   3. $x_{t+1} \sim p(\cdot | x_t, u_t)$;

4. Return $\tau = (x_0, u_0, x_1, \ldots, u_{N_T}, x_{N_T} + 1)$;

B. PAC Feedback Motion Planning

While the ability to compute PAC open-loop policies is valuable, ideally, we would like to compute closed-loop policies of the form $\pi_t(x_t, \xi)$. A closed-loop policy should enable us to generate tighter state-space trajectory distributions and improve the ability of our policies to satisfy control objectives. However, designing feedback policies is not straightforward and can be very system specific. One fair general approach
might be to select a local feedback control policy of the form
\[ u_t = K_t(x_t - x_0) + u_t^d, \]
where \( \kappa = \{K_0, K_1, \ldots, K_{N_T}\} \) is a sequence of time-varying gains, \( K_t \in \mathbb{R}^{N_x \times N_u} \). \( x_0 \in \mathbb{R}^{N_x} \) is a nominal state trajectory and \( u_t^d \in \mathbb{R}^{N_u} \) is a nominal input trajectory. All of the elements of these sequences are parameters of the feedback policy. Assuming the surrogate distribution is a multivariate Gaussian with a diagonal covariance matrix, in this case, \( \mathcal{E} \in \mathbb{R}^{2(N_xN_u + N_u + N_x)N_T} \). Given the size of the decision space, this parameterization of the policy is unlikely to be computationally tractable. To overcome this challenge, we instead use a local closed-loop policy of the form
\[ u_t = K_t(\tau^d(\mathcal{E}, x_0)) (x_t - x_0) + u_t^d(\mathcal{E}). \]
(10)

Here we refer to \( \tau^d = \{x_t^d, u_t^d, x_{t1}^d, u_{t1}^d, \ldots, x_{N_T}^d, u_{N_T}^d\} \) as the nominal trajectory, which is computed using a nominal deterministic discrete-time dynamics model, \( f(x_t^d, u_t^d) \).

\[
x_{t+1}^d = x_t^d + f(x_t^d, u_t^d)\Delta t;
\]
(11)

where \( u_t^d = \zeta_t \). To sample from this feedback policy, we must first compute the nominal trajectory, \( \tau^d \). However, \( \tau^d \) itself is dependent on \( \mathcal{E} \); for this reason, we employ a two-stage sampling process. In the first stage, we sample from the trajectory distribution \( p(\tau^d|\mathcal{E}) \) using the deterministic dynamics model (11) and the surrogate distribution \( p(\mathcal{E}|\nu) \). For each sample trajectory \( \tau^d \), we compute the time-varying gains \( K_t(\tau^d) \). In the second stage, we sample from the closed-loop stochastic dynamics using the feedback policy (10). This approach, summarized in Algorithm 4, certifies the feedback policy by allowing our sampled costs and constraints to capture the impact of our local feedback policy on our stochastic dynamics model.

We use the finite horizon, discrete, time-varying linear quadratic regulator (TVLQR) [37] to compute the feedback gains \( K_t(\tau^d) \). For each sampled nominal trajectory, the gains are computed by integrating the finite horizon discrete-time Riccati equations backward in time using the nominal dynamics linearized about \( \tau^d \).

C. PAC-NMPC

Finally, we propose a receding-horizon NMPC approach based on our PAC feedback motion planning algorithm (Algorithm 5). At each planning interval, with a period of \( H \), the trajectory distribution and corresponding feedback policy is optimized given the current state, \( x_0 \), and the prior hyper-parameters, \( \nu_0 \). Then, the trajectory and feedback gains are trimmed by \( \frac{\Delta}{2} \) to account for time passed since the beginning of the planning interval and executed by the controller.

V. SIMULATION EXPERIMENTS

A. Trajectory Optimization Experiments

We simulate a stochastic bicycle model with acceleration and steering rate inputs. We denote \( x_t = [x_0, x_1, x_2, x_3, x_4]^T \) as the state vector, \( u_t = [u_0, u_1]^T \) as the control vector, \( l = 0.33 \) as the wheel base, and \( \Gamma = \text{diag}(0.001, 0.001, 0.1, 0.2, 0.001) \) as
the covariance.

\[
\begin{align*}
    x_{t+1} &\sim p(\cdot| x_t, u_t) \triangleq x_t + (f(x_t, u_t) + \omega) \Delta t \\
    f(x_t, u_t) &= [x_3 \cos(x_2), x_3 \sin(x_2), x_3 \tan(x_4)/l, u_0, u_1] \\
    \omega &\sim \mathcal{N}(\cdot|0, \Gamma)
\end{align*}
\]  

(12)

A 20 timestep trajectory with \( \Delta t = 0.1 \) sec is optimized, with an initial state of \( x_1 = [0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] \) and a goal state of \( x_G = [3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] \). The steering angle is constrained to \( (-0.4, 0.4) \) rad, the steering rate to \( (-1.0, 1.0) \), and the acceleration to \( (-1.0, 1.0) \). We apply a quadratic cost on the final state of the trajectory: \( J(\tau) = x_G^T x_G + Q_f x_f + \Gamma \). An obstacle constraint is applied with circular obstacles at \( (1.0, 0.75) \) and \( (2.0, -0.75) \). The initial prior distribution has a mean of all zeros and a variance of all ones. Parameters were as follows: \( L = 5, M = 1024, \delta = 0.05, \gamma = 10 \). Simulations were run on a laptop with an Intel Core i9-9880H CPU and a Nvidia GeForce RTX 2070 GPU.

We ran trajectory optimization for 500 iterations. Once completed, our method returns the hyper-parameters, \( \hat{\nu} \), and the optimized PAC bounds, \( \hat{J}(\hat{\nu}) \) and \( \hat{C}(\hat{\nu}) \). It is guaranteed that policies sampled from \( p(\xi|\nu') \) will have an expected cost of \( \leq J_0(\hat{\nu}) \) and an expected constraint (probability of collision) of \( \leq C_0(\hat{\nu}) \) with a 95% chance. To validate the PAC bounds, we compare them against Monte Carlo estimates of the expected cost, \( \hat{J}(\hat{\nu}) \), and probability of collision, \( \hat{C}(\hat{\nu}) \), at each iteration.

To highlight the necessity of properly considering stochasticity to generate accurate guarantees, we compared performance while optimizing the PAC bounds with and without considering stochastic dynamics (replace line 3 of Algorithm 3 and line 9 of Algorithm 4 with \( x_{t+1} = x_t + f(x_t, u_t) \Delta t \)). Optimizing without stochastic dynamics caused \( \hat{J}(\hat{\nu}) \) and \( \hat{C}(\hat{\nu}) \) to be larger than the bounds, indicating that the guarantees were not accurate (Fig. 2(a), (b)). We also performed an ablation study in which we compared performance with and without feedback. On average, each iteration took 15.9 ms without feedback and 18.5 ms with feedback. Feedback enabled the optimizer to reduce the bounds to lower values (Fig. 2(e)–(h)) and resulted in a tighter distribution in state-space (Fig. 3).

We compare against optimizing an empirical approximation of the expected costs/constraints, regulated by the \( KC \)-divergence to prior policies, to demonstrate that optimizing the PAC bounds encourages more robust polices. We also compare against RA-MPPI [30], a state-of-the-art risk-aware sampling based NMPC method. We used the following parameters: \( \Sigma_c = 0.01, M = 1024, N = 300, \eta = 0, \alpha = 0.3, A = 10, B = 1, C_\nu = 0.5 \) for the cost, and \( C_\nu = 0.05 \) for the constraint. Our approach converged to the lowest estimated expected cost and probability of collision (Fig. 4).
The route, virtual obstacle placements, and $x$. 

\[ \delta + 1 = 0 \leq \Delta = 0 \Leftrightarrow 1024 \]

was successfully able to control the system in real time and followed the path while avoiding nearby obstacles. In Fig. 5, we show the route taken by the system during five loops of the path.

During each planning interval of $H = 0.2$ sec, the controller achieved an average of 54 iterations of PAC stochastic trajectory optimization (approximately 3.7 ms per iteration). The controller consistently produced PAC bounds $\leq 10\%$ probability of collision, which accurately bounded the Monte Carlo estimates when optimizing with stochastic dynamics. The bound was exceeded only once out of 482 planning intervals. Thus, the estimates were bounded accurately in around 99.8% of planning intervals, well within the 95% confidence bound (Fig. 6(d)).

We demonstrate a quantitative difference in the observance of the PAC Bounds in Fig. 6. When optimizing the bounds with the nominal dynamics, the probability of collision estimates were only bounded in 93.9% of planning intervals (Fig. 6(b)). This decrease in bound accuracy is even more apparent when running the controller without feedback to compensate for unmodeled noise, resulting in only 64.7% of planning intervals having bounded probability of collision estimates (Fig. 6(a)).

VI. HARDWARE EXPERIMENTS

A. Rally Car

1) Hardware: To evaluate our method on physical hardware, we ran PAC-NMPC on a 1/10th-scale Traxxas Rally Car platform. The algorithm runs on a Nvidia Jetson Orin mounted on the bottom platform. The control interface is a variable electronic speed controller (VESC), which executes commands and provides servo state information. The position and orientation is tracked with an OptiTrack system.

2) Setup: The route, virtual obstacle placements, and TVLQR costs are identical to those in the simulation experiments. We optimize a 12 timestep trajectory with $\Delta t = 0.1$ sec at a replanning period of $H = 0.2$ sec. We apply a quadratic cost to the final state of the trajectory: $J(\tau) = x_{N_T+1}^T Q_f x_{N_T+1}$ where $Q_f = diag([1.0, 1.0, 0.1, 0.1, 0.0])$. Parameters were as follows: $L = 5$, $M = 1024$, $\delta = 0.05$, $\gamma = 10$. To allow for a higher control rate, the executed control trajectory is interpolated from 10 Hz to 50 Hz with the assumption that our PAC bounds will still hold. We perform an ablation study in which we compare performance with and without feedback, as well as optimizing the PAC bounds with and without stochastic dynamics.

PAC-NMPC was successfully able to control the system in real time and followed the path while avoiding nearby obstacles. In Fig. 5, we show the route taken by the system during five loops of the path.

During each planning interval of $H = 0.2$ sec, the controller achieved an average of 54 iterations of PAC stochastic trajectory optimization (approximately 3.7 ms per iteration). The controller consistently produced PAC bounds $\leq 10\%$ probability of collision, which accurately bounded the Monte Carlo estimates when optimizing with stochastic dynamics. The bound was exceeded only once out of 482 planning intervals. Thus, the estimates were bounded accurately in around 99.8% of planning intervals, well within the 95% confidence bound (Fig. 6(d)).

We demonstrate a quantitative difference in the observance of the PAC Bounds in Fig. 6. When optimizing the bounds with the nominal dynamics, the probability of collision estimates were only bounded in 93.9% of planning intervals (Fig. 6(b)). This decrease in bound accuracy is even more apparent when running the controller without feedback to compensate for unmodeled noise, resulting in only 64.7% of planning intervals having bounded probability of collision estimates (Fig. 6(a)).
Fig. 8. Path taken while following route (black) and avoiding obstacles (brown) with rally car hardware. Blue/green paths use stochastic bounds, red/orange paths do not. Blue/orange paths use feedback, red/green paths do not. Collisions are annotated by arrows.

Fig. 9. Optimized PAC Bound compared to Monte Carlo estimates for probability of constraint violation at each planning interval. When optimizing the bound with nominal dynamics, the car collided with virtual obstacles, with and without feedback (Fig. 9(a), (b)).

Fig. 10. Edge540 acceleration noise model.

model (Fig. 9(c)). When optimizing the bound with nominal dynamics, the car collided with virtual obstacles, with and without feedback (Fig. 9(a), (b)).

B. Fixed-Wing UAV

1) Hardware: To demonstrate that our approach can extend to more complex, high-dimensional systems, we use PAC-NMPC to control a 24" wingspan Edge 540 fixed-wing UAV. It has a controllable propeller, rudder, elevator, and kinematically linked ailerons. The position and orientation are tracked with an OptiTrack system.

2) Setup: We utilize a quaternion formulation of the fixed-wing described in [2] with RK2 integration. This system has a 17-dimensional state space and a 4-dimensional control space. The state of the system is $x = [r, q, \delta, v, \omega, p]$ where $r \in \mathbb{R}^3$ is the position, $q \in Q$ is the orientation, $\delta \in \mathbb{R}^3$ are control surface deflections of the ailerons, elevator, and rudder respectively, $v \in \mathbb{R}^3$ is the linear velocity, $\omega \in \mathbb{R}^3$ is the angular velocity, and $p$ is the propeller speed. The control signal is the rate of change of the control surface deflections and of the propeller speed: $u = [u_a, u_e, u_r, u_p]$. We place noise represented by a 3 component Gaussian Mixture Model over the body frame accelerations, which was fit from 2000 samples of data collected with the fixed-wing (Fig. 10). We optimize a 12 timestep trajectory with $\Delta t = 0.1$ sec at a replanning period of $H = 0.2$. We apply a quadratic cost of the form $J(\tau) = x_{N+1}^T Q x_{N+1} + \sum_{t=1}^{N} (x_t^T Q x_t + u_t^T R u_t)$ and an obstacle constraint. Parameters were as follows: $L = 1$, $M = 1024$, $\delta = 0.05$, and $\gamma = 10$.

We use feedback and use stochastic dynamics to optimize the PAC bounds. To enable real-time performance, we set the number of prior policies, $L$, to one and compute the linearized dynamics for the feedback policy using finite difference on the mean trajectory of the prior. The controller was run on a desktop computer with an AMD Ryzen 7 5800x and a NVIDIA GeForce RTX 3080.

3) Results: PAC-NMPC was successfully able to control the fixed-wing in real-time, following a square path while avoiding nearby obstacles. In Fig. 11, we display the route taken by the system during five loops of the path. During each planning interval of $H = 0.2$ sec, the controller achieved an average of 22 iterations of trajectory distribution optimization with around

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
8.5 ms per iteration. The controller consistently produced PAC bounds of around 10% probability of collision, which accurately bounded the Monte Carlo estimates (Fig. 12).

VII. DISCUSSION

In this work, we presented a novel SNMPC method capable of propagating uncertainty through arbitrary nonlinear dynamic systems and of providing statistical guarantees on expected cost and constraint violations. We demonstrated real time performance both in simulation and on-board physical hardware. Further, we show that the algorithm is capable of scaling to more complex systems, like fixed-wing UAVs. Since this algorithm can be used with “black-box” sampling of dynamics (assuming they are continuously differentiable), costs, constraints, and noise, future work can investigate its use with learned dynamics and perception-informed costs. Future work could also explore the extension of this approach to more complex control trajectory distributions.

REFERENCES

[1] D. Falanga, P. Feohn, P. Lu, and D. Scaramuzza, “PAMPC: Perception-aware model predictive control for quadrotors,” in Proc. IEEE/RSJ Int. Conf. Intel. Conf. Robot. Syst., 2018, pp. 1–8.

[2] M. Basescu and J. Moore, “Direct NMPC for post-stall motion planning with fixed-wing UAVs,” in Proc. IEEE Int. Conf. Robot. Automat., 2020, pp. 9592–9598.

[3] Y. Ding, A. Pandala, and H.-W. Park, “Real-time model predictive control for versatile dynamic motions in quadrupedal robots,” in Proc. IEEE Int. Conf. Robot. Automat., 2019, pp. 8484–8490.

[4] R. Granda, F. Farshidian, R. Rantil, and M. Hutter, “Feedback MPC for torque-controlled legged robots,” in Proc. IEEE/RSJ Int. Conf. Intel. Robots Syst., 2019, pp. 4730–4737.

[5] J. A. Paulson and A. Mesbah, “An efficient method for stochastic optimal control with joint chance constraints for nonlinear systems,” Int. J. Robust Nonlinear Control, vol. 29, no. 15, pp. 5017–5037, 2019.

[6] Z. Manchester and S. Kuindersma, “DIRTREL: Robust trajectory optimization with ellipsoidal disturbances and LQR feedback,” in Proc. IEEE Robot. Sci. Syst., 2017, doi: 10.15607/RSS.2017.XIII.057.

[7] S. Singh, A. Majumdar, J.-J. Slotine, and M. Pavone, “Robust online motion planning via contraction theory and convex optimization,” in Proc. IEEE Int. Conf. Robot. Automat., 2017, pp. 5883–5890.

[8] N. Ozaki, S. Campagnola, and R. Funase, “Tube stochastic optimal control for nonlinear constrained trajectory optimization problems,” J. Guid. Control, Dyn., vol. 43, pp. 645–655, 2020.

[9] J. Yin, Z. Zhang, E. Theodorou, and P. Tsiofaris, “Trajectory distribution control for model predictive path integral control using covariance steering,” in Proc. IEEE Int. Conf. Robot. Automat., 2022, pp. 1478–1484.

[10] M. Kohlharov, “Sample complexity bounds for iterative stochastic policy optimization,” in Proc. Adv. Neural Inf. Process. Syst., 2015, pp. 3114–3122.

[11] M. Shechells, “Probably approximately correct robust policy search with applications to mobile robotics,” Ph.D. dissertation, Johns Hopkins Univ., Baltimore, MD, USA, 2020.

[12] M. E. Villaneuva, R. Quirynen, M. Diehl, B. Chachuat, and B. Houska, “Robust MPC via min-max differential inequalities,” Automatica, vol. 77, pp. 311–321, 2017.

[13] G. Garimella, M. Shechells, J. L. Moore, and M. Kohlharov, “Robust obstacle avoidance using tube NMPC,” in Proc. Robotics: Sci. Syst., 2018, doi: 10.15607/RSS.2018.XIV.055.

[14] S. Lucia, R. Paulsen, and S. Engell, “Multi-stage nonlinear model predictive control with verified robust constraint satisfaction,” in Proc. IEEE 53rd Conf. Decis. Control, 2014, pp. 2816–2821.

[15] D. Q. Mayne, E. C. Kerrigan, E. Van Wyk, and P. Falugi, “Tube-based robust nonlinear model predictive control,” Int. J. Robust Nonlinear Control, vol. 21, no. 11, pp. 1341–1353, 2011.

[16] D. L. Marruedo, T. Alamo, and E. F. Camacho, “Input-to-state stable MPC for constrained discrete-time nonlinear systems with bounded ad-\n
utainties,” in Proc. IEEE 41st Conf. Decis. Control, 2002, pp. 4619–4624.

[17] Y. Gao, A. Gray, H. E. Tseng, and F. Borrelli, “A tube-based robust nonlinear predictive control approach to semi-autonomous ground vehicles,” Veh. Syst. Dyn., vol. 52, no. 6, pp. 802–823, 2014.

[18] F. Bauer, M. Bürger, and F. Allgöwer, “Discrete-time incremental ISS: A framework for robust NMPC,” in Proc. Eur. Control Conf., 2013, pp. 2068–2073.

[19] G. Chou, N. Ozay, and D. Berenson, “Model error propagation via learned contraction metrics for safe feedback motion planning of unknown systems,” in Proc. IEEE 60th Conf. Decis. Control, 2021, pp. 3576–3583.

[20] J. Köhler, R. Soloperto, M. A. Müller, and F. Allgöwer, “A computationally efficient robust model predictive control framework for uncertain nonlinear systems,” IEEE Trans. Autom. Control, vol. 66, no. 2, pp. 794–801, Feb. 2021.

[21] T. Lew and M. Pavone, “Sampling-based reachability analysis: A random set theory approach with adversarial sampling,” in Proc. Conf. Robot. Learn., 2021, pp. 2055–2070.

[22] A. Wu, T. Lew, K. Solovey, E. Schmerling, and M. Pavone, “Robust- RRT: Probabilistically-complete motion planning for uncertain nonlinear systems,” in Proc. Int. Symp. Robot. Res., 2023, pp. 538–554.

[23] E. Todorov and W. Li, “A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems,” in Proc. Amer. Control Conf., 2005, pp. 300–306.

[24] E. Todorov and Y. Tassa, “Iterative local dynamic programming,” in Proc. IEEE Symp. Adaptive Dyn. Program. Reinforcement Learn., 2009, pp. 90–95.

[25] E. Theodorou, Y. Tassa, and E. Todorov, “Stochastic differential dynamic programming,” in Proc. IEEE Int. Conf. Robot. Automat., 2010, pp. 1125–1132.

[26] T. A. Howell, C. Fu, and Z. Manchester, “Direct policy optimization using deterministic sampling and collocation,” IEEE Robot. Automat. Lett., vol. 6, no. 3, pp. 5324–5331, Jul. 2021.

[27] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, “Aggressive driving with model predictive path integral control,” in Proc. IEEE Int. Conf. Robot. Automat., 2016, pp. 1433–1440.

[28] G. Williams, B. Goldfain, P. Drews, K. Saigol, J. M. Rehg, and E. A. Theodorou, “Robust sampling based model predictive control with sparse objective information,” in Proc. Robot. Sci. Syst., 2018, doi: 10.15607/RSS.2018.XIV.042.

[29] M. S. Gandhi, B. Vlahov, J. Gibson, G. Williams, and E. A. Theodorou, “Robust model predictive path integral control: Analysis and performance guarantees,” IEEE Robot. Automat. Lett., vol. 6, no. 2, pp. 1423–1430, Feb. 2021.

[30] J. Yin, Z. Zhang, and P. Tsiotras, “Risk-aware model predictive path integral control using conditional value-at-risk,” in Proc. IEEE Int. Conf. Robot. Automat., 2023, pp. 7937–7943.

[31] M. P. Deisenroth et al., “A survey on policy search for robotics,” Found. Trends Robot., vol. 2, no. 1/2, pp. 1–142, 2013.

[32] A. Majumdar, A. Farid, and A. Sonar, “PAC-Bayes control: Learning policies that provably generalize to novel environments,” Int. J. Robot. Res., vol. 40, no. 2/3, pp. 574–593, 2021.

[33] S. Veer and A. Majumdar, “Probably approximately correct vision-based planning using motion primitives,” in Proc. Conf. Robot. Learn., 2021, pp. 1001–1014.

[34] J. Peters, K. Mulling, and Y. Altun, “Relative entropy policy search,” in Proc. AAAI Conf. Artif. Intell., 2010, pp. 1607–1612.

[35] O. Catoni, “Challenging the empirical mean and empirical variance: A deviation study,” Annales de l’IHP Probabilité et statistiques, vol. 48, no. 4, pp. 1148–1185, 2012.

[36] Y. Fei, G. Rong, B. Wang, and W. Wang, “Parallel l-BFGS-B algorithm on GPU,” in Proc. Europ. Conf. Comput. Graph., vol. 40, pp. 1–9, 2014.

[37] R. Tedrake, “Underactuated robotics,” 2022. [Online]. Available: http://underactuated.mit.edu