2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

LP-MPPI: Low-Pass Filtering for Efficient Model Predictive Path Integral Control

Piotr Kicki

***Abstract*— Model Predictive Path Integral (MPPI) control is a widely used sampling-based approach for real-time control, valued for its flexibility in handling arbitrary dynamics and cost functions. However, it often suffers from high-frequency noise in the sampled control trajectories, which hinders the search for optimal controls and transfers to the applied controls, leading to actuator wear. In this work, we introduce Low-Pass Model Predictive Path Integral Control (LP-MPPI), which integrates low-pass filtering into the sampling process to eliminate detrimental high-frequency components and enhance the algorithm's efficiency. Unlike prior approaches, LP-MPPI provides direct and interpretable control over the frequency spectrum of sampled control trajectory perturbations, leading to more efficient sampling and smoother control. Through extensive evaluations in Gymnasium environments, simulated quadruped locomotion, and real-world F1TENTH autonomous racing, we demonstrate that LP-MPPI consistently outperforms state-of-the-art MPPI variants, achieving significant performance improvements while reducing control signal chattering.**

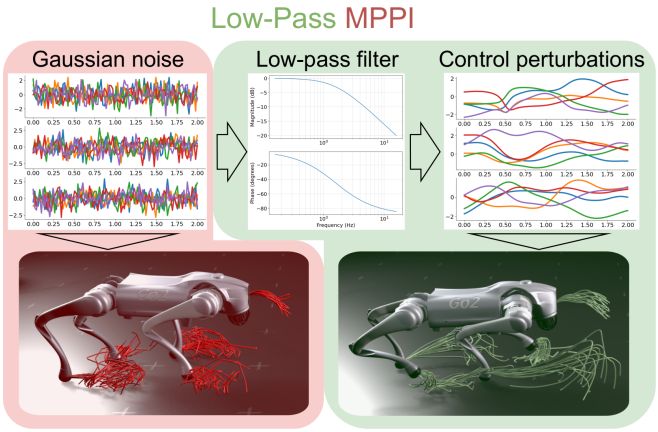

Fig. 1. The proposed Low-Pass Model Predictive Path Integral (LP-MPPI) control smooths out the controls via the low-pass filtering of control perturbations and increases the efficiency of the MPPI.

I. INTRODUCTION

One of the key abilities of the autonomous system is to determine the best actions given a certain goal, i.e., real-time motion planning and control. If the model of the controlled system is available, one of the best performing approaches is Model Predictive Control (MPC), which has proven its capabilities to solve many challenging tasks, such as autonomous racing [1], off-road driving [2], agile drone flight [3], and legged locomotion [4].

In general, there are two main approaches to MPC: sampling- and optimization-based. Optimization-based MPC algorithms provide an efficient way to find optimal control sequences using dynamics and cost function gradients [5], [6]. However, they typically impose requirements on the dynamics model or cost function formulations, such as differentiability or continuity. An interesting alternative is sampling-based MPC. One of the main benefits of this approach is that the dynamics and cost functions can be arbitrary, and the only requirement is to evaluate them relatively fast. The two algorithms within this group, which have proven their effectiveness in numerous tasks, such as off-road driving [2], drone flight [7], and control of high-dimensional simulated systems (humanoids, dexterous hands, manipulators) [8], [9], are the Cross Entropy Method (CEM) [10] and Model Predictive Path Integral Control (MPPI) [2].

Piotr Kicki is with Institute of Robotics and Machine Intelligence, Poznan University of Technology, Poznan, Poland and with IDEAS Research Institute, Warsaw, Poland piotr.kicki@put.poznan.pl

This work was supported by the National Science Centre, Poland, under the INTENTION project (grant no. 2021/43/I/ST6/02711), PUT internal grant 0214/SBAD/0256, and by the infrastructure of the Poznan Supercomputing and Networking Center.

The core of both MPPI and CEM approaches is to evaluate the performance of the control trajectories sampled from a sequence of Gaussian distributions. This approach results in a potential lack of temporal correlation within individual trajectories (see the red part of Fig. 1). In fact, control trajectory perturbations are approximately white noise signals, i.e., they are evenly composed of all possible frequencies. This, in turn, results in the overrepresentation of the high-frequency components when compared to the expected optimal behaviors in most robotic systems. Moreover, the dynamics of most robots naturally dampen these components, so their impact on the resulting performance is minimal. Thus, they do not contribute to the efficient search for optimal control signals but instead result in chattering of the applied controls and wear out of the actuators [11].

To address these issues, researchers proposed several interesting approaches, such as spline interpolation of control signals [9], [12] or input-lifting [13]. However, they offer only an indirect and coarse control over the control signal frequency spectrum. A more direct approach, inspired by the iCEM algorithm [8], was proposed in [14], where colored noise was used in MPPI instead of the white one. However, colored noise damps the signal components proportionally to the inverse of their frequency, or their powers, which may not be desired, especially for tasks that require frequent repetitiveness, e.g., legged locomotion, stirring, or chopping.

In this paper, we address these issues by extending the MPPI algorithm with a low-pass filter applied to the sampled control perturbations. Our method, Low-Pass Model Predictive Path Integral control (LP-MPPI), constrains exploration to a task-relevant frequency band by filtering out high-

frequency noise in the control perturbations (see Fig. 1). This improves sample efficiency and produces smoother control signals without compromising responsiveness. Moreover, unlike colored noise, it does not bias the control signal frequencies in the filter's passband, enabling an effective search for high-performing trajectories across the admissible frequency range. Our method introduces only two interpretable parameters with clear physical meaning, allowing intuitive tuning. They directly control the frequency range in which the search for the optimal control trajectory is focused and the damping characteristics beyond it. Finally, LP-MPPI is easy to implement and adds negligible computational overhead compared to the MPPI algorithm.

We conduct an extensive experimental evaluation of the proposed method against the State-of-the-Art variants of the MPPI in three Gymnasium environments, over a wide range of MPPI parameters. The results demonstrate that our approach consistently outperforms the considered baselines, improving their results by 24% on average, while significantly reducing chattering in the applied control signals. To further assess its applicability, we integrate LP-MPPI with the recently developed Dial-MPC [15] and evaluate the resulting LP-Dial-MPC on quadruped robots. This experiment highlights the ease and effectiveness with which our method can be combined with other MPPI-based approaches, yielding an average performance gain of over 32% compared to Dial-MPC. Finally, we validate our approach in real-world autonomous F1TENTH racing, where it outperforms all baselines, most of them by a large margin.

Our contributions can be summarized as follows:

- We propose a Low-Pass Model Predictive Path Integral algorithm, which enables shaping the frequency spectrum of the control trajectories' perturbations distribution, improving search efficiency and reducing the high-frequency noise in the applied control signal.
- We conduct a thorough experimental analysis of the proposed approach in several simulated environments, showing that the proposed method consistently outperforms the state-of-the-art MPPI-based control approaches, while reducing the amount of high-frequency components in the applied control signals (see Tab. I).
- We highlight the practicality of the proposed solution, that is, the ease of implementation of our method and its integration with the latest sampling-based MPC methods, the physical interpretability of its parameters, and the intuitiveness of tuning them.

II. Related Work

In recent years, numerous works have sought to enhance the effectiveness of the MPPI algorithm [9], [12], [13], [14], [15]. Many of these approaches focus on improving control smoothness and increasing sampling efficiency.

Both in [13] and [16], authors proposed to smooth the control signals generated by MPPI, by (i) lifting the input-space, i.e., sampling in the space of the derivatives of the actual controls, and (ii) adding a smoothness cost. This approach effectively transforms the frequency spectrum of the control signals' perturbations into colored noise and gives only minimal control of its shape. Our frequency-based shaping of control signal perturbations can adjust the time correlation of the trajectories in a more fine-grained way, does not introduce such a strong bias to the lowest frequencies, and does not require additional cost terms. Moreover, adding smoothness costs reduces search efficiency because some of the drawn trajectories are effectively excluded due to high smoothness costs.

Interestingly, the authors of [13] showed that using a low-pass Savitzky-Golay filter on the control trajectory update signal (see MPPI(SGF) in [13]) results in a poorer performance than their proposed SMPPI. Our work shows that if the low-pass filtering is applied before the control perturbations are simulated, it maintains constraint satisfaction, introduces no phase distortion, and outperforms SMPPI [13].

Another interesting strategy to smooth the control signals and improve exploration efficiency is to exploit spline interpolation [9], [12], [17]. Instead of sampling a long sequence of actions, one can sample only a reduced number of spline control points and then interpolate between them. Thus, the dimension of the exploration space is reduced, and the resulting trajectories are smoother. However, in this approach, the control over the smoothness is indirect and rather coarse and may require predicting the time increments along the controls [17], [18].

A more direct way to improve control smoothness and exploration is to modify the control perturbation sampling distribution itself, since the default Gaussian choice may not be an optimal choice. In this theme, the authors of [19] propose to use a mixture of the original normal distribution with the log-normal distribution to increase the probability of sampling larger deviations from the nominal control sequence to improve exploration. In turn, in [20] and [21], authors explored the use of multimodal distributions, i.e., Gaussian Mixture Models and empirical particle-based distributions with Stein Variational Gradient Descent updates, to increase the exploration capabilities by allowing one to maintain multiple hypotheses. Recently, learning-based approaches have also been used to determine better sampling distributions based on the structure of the environment [22], [23], enabling focusing the search on promising areas of the state space. Nevertheless, none of these methods considers the correlation between subsequent controls within each sampled control trajectory perturbation, which may result in a chattering of the control signal.

Methods that adjust the sampling distribution to address this particular issue, while increasing the exploration capabilities of the sampling-based MPC, were proposed in [8] and [14]. The approach presented in [8], called iCEM, is an improved version of the Cross-Entropy Method (CEM) [10]. It shows that sampling the actions from the colored-noise distributions significantly improves the original CEM approach. The authors of [14] employed the same idea in the MPPI algorithm, demonstrating improved smoothness and performance. In our work, we also focus on shaping the MPPI sampling distribution in the frequency domain. How-

ever, instead of relying on colored noise, we use a low-pass filter, which provides greater flexibility and enables uniform sampling across the desired range of low frequencies.

III. Method

A. Problem definition

We consider the following optimal control problem

$$\min_{u_{t:t+H}} J(x_{t:t+H+1}, u_{t:t+H}) = \sum_{h=0}^{H} c(x_{t+h}, u_{t+h}) + c_f(x_{t+H+1})$$
$$\text{subject to} \quad (1)$$
$$x_{t+h+1} = f(x_{t+h}, u_{t+h}), \quad \forall h \in \{0, ..., H\}$$
$$x_{t:t+H+1} \in \mathcal{X}, u_{t:t+H} \in \mathcal{U},$$

where x_t and u_t are the state and control signal at time t, H is optimization horizone, f is the system dynamics, c and c_f are step and terminal cost functions, J is the total cost of the trajectory, while $\mathcal{X}$ and $\mathcal{U}$ are the sets of admissible states and controls, respectively.

B. Model Predictive Path Integral Control

Our proposed method, Low-Pass Model Predictive Path Integral Control (LP-MPPI), is based strongly on the original MPPI algorithm [2]. Therefore, we recall its pseudocode in Algorithm 1. In general, the idea is to (i) draw multiple control trajectories around the current nominal control trajectory, (ii) simulate them using the model of the system, (iii) compute their costs, and finally (iv) update the nominal control trajectory based on the costs obtained by the perturbed controls. In this paper, we focus on the commonly overlooked aspect of the MPPI algorithm – drawing random control perturbations. In fact, the only part of the algorithm that we would like to analyze and improve is located in line 2. To do so, we will analyze the original MPPI algorithm [2] and its recent extension [14] from a frequency domain perspective.

Algorithm 1 Model Path Integral Control (MPPI)

Require: Nominal control sequence U, current state x_t, System dynamics model f, step and terminal cost functions c, c_f, number of rollouts N, control sequence horizon H, temperature parameter λ, noise covariance matrix Σ

Ensure: Updated nominal control sequence U

1: **for** $i = 1$ to N **do** ▷ Sample N trajectories
2: Sample noise sequence $\epsilon_{i,1:H}$ from $\mathcal{N}(0, \Sigma)$
3: Generate control sequence $U_i = U + \epsilon_i$
4: Compute trajectory $x_{i,t:t+H+1}$ using system dynamics $x_{i,t+h+1} = f(x_{i,t+h}, u_{i,h})$
5: Compute cost $J_i = J(x_{i,t:t+H+1}, u_{i,1:H})$
6: **end for**
7: Compute importance weights $w_i = \frac{e^{-\lambda J_i}}{\sum_{j=1}^{N} e^{-\lambda J_j}}$
8: Update controls $U = \sum_{i=1}^{N} w_i U_i$
9: Apply first control input u_1 to the system
10: Shift control sequence: $U \leftarrow \{u_2, ..., u_H, u_{H+1}\}$

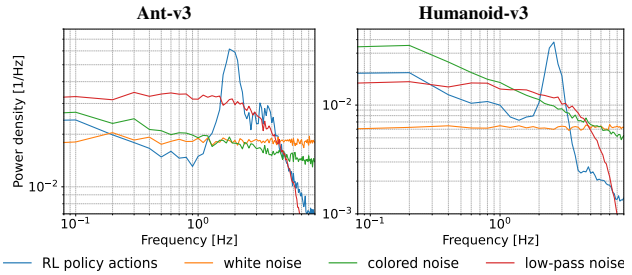

Fig. 2. Spectrograms of the actions drawn from the trained RL policy and the different sampling distributions. Both white and colored noise are unable to closely fit the spectrum of RL behaviors, while the low-pass filtered noise covers it quite accurately.

C. Spectral analysis

In the original MPPI, noise sequences are sampled from a Gaussian distribution with a constant covariance matrix Σ, resulting in temporally uncorrelated (white) perturbations. In general, this can be advantageous, as white noise is widely used in system identification [24] to excite dynamics across all control frequencies. In MPC, a similar argument applies to the cost function J, which depends on both controls and state trajectories. However, in robotic applications, exciting high-frequency modes is rarely necessary, since most robots naturally attenuate them. Therefore, control trajectories with high-frequency noise and low-frequency noise may yield similar costs, which may result in frequent changes of the following actions, and thus chattering in the real system. Furthermore, optimal control trajectories in robotics typically occupy a limited bandwidth rather than the entire frequency spectrum. Consequently, sampling uncorrelated noise amounts to maximal spectral exploration, which can reduce search efficiency and degrade performance.

To limit the amount of high-frequency noise in the sampled controls, the authors of [14] proposed biasing the spectrum of the control trajectory perturbations towards low frequencies. As a result, they obtained a strategy that puts most of the signal energy in the lowest possible frequencies and gradually reduces the energy of the higher-frequency components. In turn, in this paper, we analyze an alternative approach based on low-pass filtered noise, which balances the bias toward lower frequencies with efficient exploration of the admissible spectrum, and enables explicit control of this trade-off through adjustable parameters – cutoff frequency and filter order.

To systematically compare the aforementioned control sampling strategies and justify our proposed approach, we conducted a spectral analysis experiment. Specifically, we compared the power densities of (i) white noise, (ii) colored noise, and (iii) low-pass filtered white noise with the spectrum of control signals generated by a trained RL policy from the StableBaselines3-zoo library [25] for two environments, Ant-v3 and Humanoid-v3. We selected the RL policy as a reference because its control signals can be regarded as a proxy for near-optimal behavior in these tasks, and thus reflect realistic frequency statistics of effective control trajectories. To present the results in a clean and compact

form, frequency spectra were averaged across joints and 100 seeds. For fairness, the parameters of each analyzed sampling distribution were tuned to minimize the norm between its spectrum and the RL action spectrum. The results of this comparison are presented in Fig. 2.

The results, shown in Fig. 2, reveal that the RL agent exhibits little energy at high frequencies. However, concentrating the signal energy exclusively at the lowest frequencies also leads to a mismatch, failing to capture distinct spectral peaks occurring around 2 Hz. Among the considered strategies, low-pass filtered white noise provides the closest match to the RL spectrum, as it attenuates only the unnecessary high-frequency components while preserving energy in the most relevant band. From the perspective of MPPI, this reduces sampling in dynamically suppressed areas and concentrates rollouts in regions that meaningfully affect the cost without excessively biasing the search toward the lowest frequencies, thereby improving effective sample efficiency.

D. Low-pass Model Predictive Path Integral Control

Following the observations made in the previous section, we introduce the Low-pass Model Predictive Path Integral Control (LP-MPPI). The core of the proposed algorithm is the introduction of a temporal correlation between the subsequent control perturbations by using a low-pass filter on them. The proposed approach is formalized in Algorithm 2, where the changes w.r.t. the original MPPI algorithm are marked in blue. Our method introduces a subtle yet important modification that (i) biases the search for optimal control trajectories toward low-frequency signals and (ii) allows the user to control the frequency spectrum in which the algorithm searches for candidate trajectory updates. These features can be precisely controlled by tuning the parameters of the low-pass filter, which we describe in detail in Section III-E. Importantly, the proposed modifications do not introduce significant computational overhead to the MPPI algorithm, as filtering typically requires a small number of multiplications proportional to the horizon length H and, in most cases, is significantly cheaper than evaluating the system's dynamics.

A key design choice is to decide which signals should be smoothed in order to improve efficiency and reduce chattering without sacrificing controller reactiveness or violating system constraints. In the proposed method, we apply low-pass filtering to sampled noise sequences, i.e., control trajectory perturbations (see line 2 of Algorithm 2). This strongly biases the search toward low-frequency trajectories without directly constraining the frequency content of the applied control signal. The first control input can still change rapidly, while subsequent inputs are correlated, preserving reactiveness while guiding exploration. Unlike post-update filtering methods such as MPPI(SGF) [13], our approach evaluates only bandwidth-limited, constraint-compliant perturbations and does not introduce phase shifts in the executed control. Finally, in importance-sampling terms, restricting the sampling bandwidth increases the overlap between the proposal distribution and the distribution of low-cost trajectories, thereby improving effective sample efficiency.

Algorithm 2 Low-pass Model Path Integral Control (LP-MPPI)

Require: Nominal control sequence U, current state x_t, System dynamics model f, step and terminal cost functions c, c_f, number of rollouts N, control sequence horizon H, temperature parameter λ, noise covariance matrix Σ, low-pass filter cutoff frequency f_c and order o_{LPF}

Ensure: Updated nominal control sequence U

1: **for** $i = 1$ to N **do** ▷ Sample N trajectories
2: Sample noise sequence $\epsilon_{i,1:H}$ from $\mathcal{N}(0, \Sigma)$
3: Filter noise sequence $\epsilon_i^{\text{LP}} = \text{LPFilter}(\epsilon_i, f_c, o_{\text{LPF}})$ using low-pass filter
4: Generate control sequence $U_i = U + \epsilon_i^{\text{LP}}$
5: Compute trajectory $x_{i,t:t+H+1}$ using system dynamics $x_{i,t+h+1} = f(x_{i,t+h}, u_{i,h})$
6: Compute cost $J_i = J(x_{i,t:t+H+1}, u_{i,1:H})$
7: **end for**
8: Compute importance weights $w_i = \frac{e^{-\lambda J_i}}{\sum_{j=1}^{N} e^{-\lambda J_j}}$
9: Update controls $U = \sum_{i=1}^{N} w_i U_i$
10: Apply first control input u_1 to the system
11: Shift control sequence: $U \leftarrow \{u_2, ..., u_H, u_{H+1}\}$

In our implementation, we employ a digital Butterworth filter [26]. Its maximally flat passband avoids biasing specific frequencies within the desired range. Although its roll-off is less steep than that of, e.g., Chebyshev filters, Fig. 2 suggests that sharper attenuation is not required for sampling high-performing control trajectories. Moreover, the relatively large phase shift introduced by the Butterworth filter is not problematic in our setting, since it is applied to temporally uncorrelated perturbations.

E. Parameters of LP-MPPI

Our proposed algorithm, LP-MPPI, does not introduce any additional parameters beyond those of a low-pass filter. Therefore, in our case, the parameters of the LP-MPPI are the parameters of the Butterworth filter itself – cutoff frequency f_c and order o_{LPF}. The cutoff frequency controls the width of the passband, which affects the range of frequencies with the highest power density in the drawn control signal perturbations. A lower f_c results in a greater bias at lower frequencies and an increased use of low-frequency signals. In turn, a higher f_c reduces the bias on low frequencies and allows searching in a wider range of signals. The second parameter of the Butterworth filter is its order o_{LPF}, which controls the slope of the power density spectrum in the stopband – roll-off, i.e., how much the attenuation of the signal components grows with the growth of the frequency. The higher the order, the lesser the exploration outside the passband, and the greater the bias towards low frequencies. To better visualize these dependencies, in Fig. 3 we present the impact of the filter cutoff frequency f_c and its order o_{LPF} on the magnitude of the frequency response and on the signals in the time domain.

Last but not least, one of the benefits of the proposed

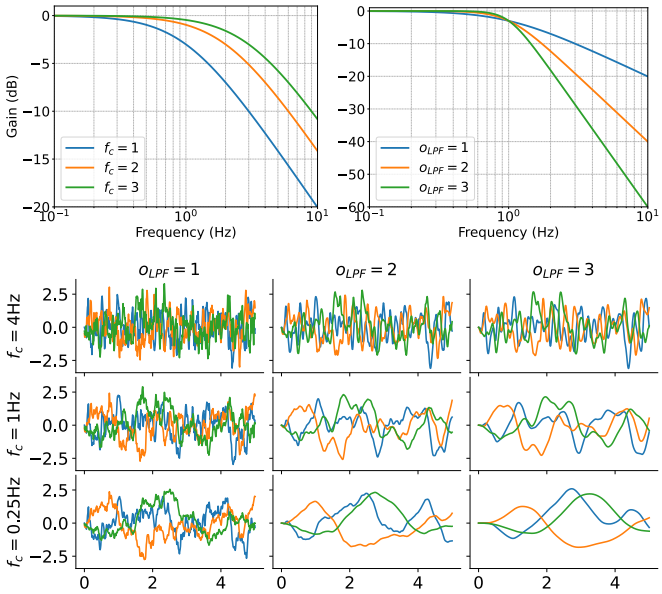

Fig. 3. Frequency response of the Butterworth filter used in the LP-MPPI algorithm (top row) and 3 example control trajectories generated by it (bottom row) for a range of cutoff frequencies f_c and filter orders o_{LPF}.

approach is the interpretability of the cutoff frequency f_c parameter, as it enables intuitive tuning. One of the natural approaches to choosing this parameter is to set it around the natural cutoff frequency of the system to be controlled with LP-MPPI, as inducing higher frequencies requires a significant effort to overcome the damping characteristics of the system itself. Moreover, one can often heuristically estimate the highest expected frequency of the system states necessary to maximize the reward or obtain the desired behavior. Furthermore, the interpretability of the cutoff frequency may help regularize the system's behavior, e.g., preventing exploitation of a carelessly designed reward function.

To sum up, the proposed LP-MPPI approach filters out the high-frequency components from the evaluated control signals within the MPPI loop, significantly reducing the high-frequency components in the obtained controls. This reduction is particularly important for robotic systems, as it improves control signals smoothness and biases the search for high-performance control signals towards a more promising area (consider the plausibility of finding a performant control signal in the upper left and bottom right corners of the bottom of Fig. 3). Finally, our approach enables one to effectively control the frequency spectrum of the control signal perturbations, while providing intuitive tuning.

IV. Experiments

A. Gymnasium environments

Our first experiment is an optimal control task in 3 high-dimensional Gymnasium environments [27]: *Hopper-v5*, *Ant-v5*, and *HalfCheetah-v5* (see Fig. 4). The goal of this experiment is to evaluate the efficiency of the proposed approach and relate it to the State-of-the-Art MPPI-based control algorithms. We compared our proposed method with the following baselines:

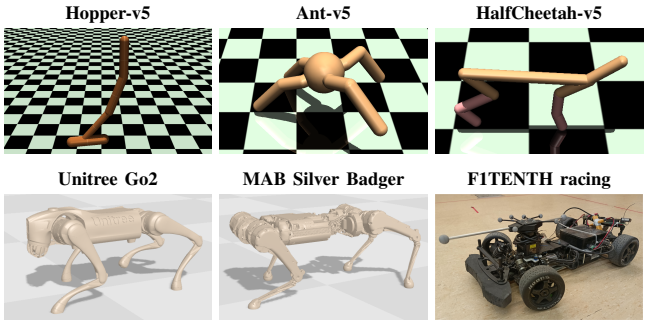

Fig. 4. Systems used in the experimental evaluation.

- MPPI – the original MPPI algorithm [2],
- SCP-MPPI – Spline-interpolated MPPI [12],
- SMPPI – smoothed MPPI by control space lifting [13],
- ColoredMPPI – MPPI with colored noise [14].

In this experiment, we assumed perfect knowledge of the system dynamics, i.e., we used the same simulated environments for MPPI rollouts and evaluation. As a cost function, we used the original rewards from the Gymnasium environments, taken with a minus sign. To ensure a fair comparison, we tuned the parameters of each algorithm using Optuna [28] and 100 trials. In addition, we averaged the results over 100 episodes to reduce the impact of randomness.

The results of our experiment are presented in Fig. 5. We evaluated each method, in each environment, on a matrix of the common MPPI parameters, i.e., horizon length H (vertical axis), and number of rollouts N (horizontal axis). In the first column, one can see the rewards obtained by the LP-MPPI, while in the remaining columns, the relative improvement of the LP-MPPI over the baselines. The superiority of the proposed approach can be seen by the dominance of the green color in the presented chart. In fact, except for very short horizons, LP-MPPI outperforms all baselines in all environments considered, no matter the number of rollouts. The scale of the improvement varies for different horizons and numbers of rollouts, but in general, the longer the horizon, the bigger the improvement. For *Ant-v5* and *HalfCheetah-v5* environments, we observe that improvements are larger for a smaller number of evaluated rollouts, indicating that LP-MPPI improves effective sample efficiency by concentrating exploration within a task-relevant frequency band. Interestingly, the biggest improvements are observed w.r.t. SMPPI and SCP-MPPI approaches, on average 26% and 36%, which may be caused by the lack of fine-grained control due to action representation. The best performance among the baselines is observed for ColoredMPPI [14], which is still, on average, about 10% less performant than the proposed LP-MPPI. In turn, if we consider the results obtained for the best pair of horizon length and number of rollouts, then LP-MPPI outperforms it on average by 8.46%. It should be noted that the rewards defined in these environments do not take into account the control smoothness; therefore, the improvement in rewards suggests that the proposed approach affects the performance of the control trajectory search.

Despite higher rewards, an important aspect of the control

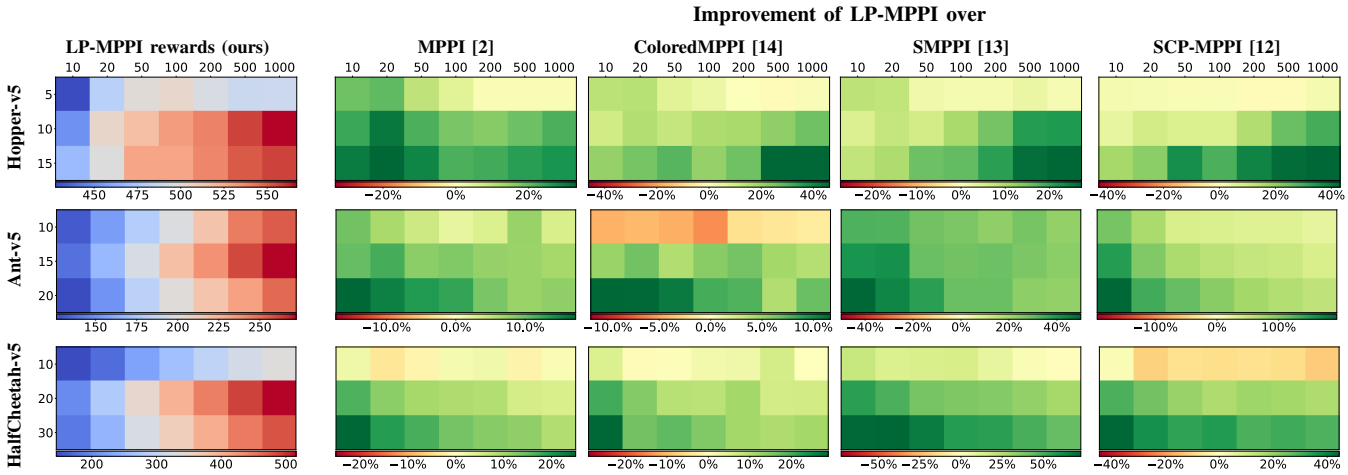

Fig. 5. Performance of the proposed LP-MPPI algorithm in Gymnasium environments (left), for a range of horizon lengths H (y-axis) and numbers of rollouts N (x-axis), and comparison to baselines (right). The green color represents the situation in which the LP-MPPI outperforms the baseline.

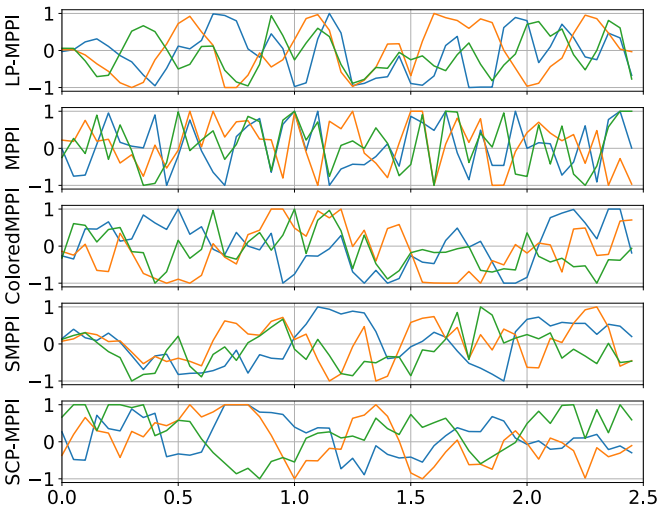

Fig. 6. Three example applied control signals in the Ant-v5 environment for the considered MPPI-based control algorithms.

algorithm is the smoothness of the applied controls. To visualize the differences between the considered methods, we present the applied control trajectories for the Ant-v5 environment with $H = 15$ and $N = 100$ in Fig. 6. For fairness of comparison, each algorithm was used with the optimal set of parameters, w.r.t. cumulative reward, determined with Optuna [28] using 100 trials. One can see that the original MPPI algorithm generates undesirably sharp controls, while the remaining baselines produce significantly smoother controls, comparable to each other and to our proposed LP-MPPI. To quantitatively assess smoothness, we report Mean Squared Second Derivative (MSSD) and Mean Savitzky-Golay Filter Deviation (MSGFD) in Table I. The results indicate that our method attains very low MSSD and the smallest MSGFD, confirming that the control trajectories it generates are notably smooth.

B. Quadruped locomotion

In the previous experiment, we considered control of simplified abstract robots. Instead, in this one, we focus on

TABLE I

COMPARISON OF CONTROL SIGNALS SMOOTHNESS IN ANT-V5 ENVIRONMENT USING THE MEAN SQUARED SECOND DERIVATIVE (MSSD) AND MEAN SAVITZKY-GOLAY FILTER DEVIATION (MSGFD).

Metric	**LP-MPPI**	MPPI	ColoredMPPI	SMPPI	SCP-MPPI
MSSD	0.409	2.144	0.805	0.402	**0.350**
MSGFD	**0.018**	0.177	0.059	0.028	0.027

the full-scale simulated quadrupeds with 12DoF. We consider a locomotion task with trot gait, imposed in the reward function to guide the search of the control signals, on two quadrupeds, i.e., Unitree Go2 and MAB Silver Badger (see Fig. 4).

Recently, it was shown that MPPI-based approaches can succeed in such complex control tasks by diffusion-style annealing of the standard deviation of the noise distribution and the use of spline interpolation, as was done in the Dial-MPC approach [15]. We aim to evaluate whether the proposed low-pass filtering can be applied to the state-of-the-art MPPI-based approach (LP-Dial-MPC) and can improve its performance. Moreover, we would like to compare our method in this setting with the best-performing approach from the previous experiment – ColoredMPPI [14]. Thus, we extend the Dial-MPC with colored noise instead of the default white one (Colored Dial-MPC).

In this experiment, we used the default settings of the Unitree Go2 trot experiment available in the code repository associated with the Dial-MPC paper [15]. The goal is to follow the desired longitudinal velocity of 1 m/s with the center of the robot trunk while maintaining its default orientation and height above the ground. In addition, a cost function that imposes a specific foot-height trajectory encourages the robot to follow a trot gait. Moreover, we designed a very similar experiment for the MAB Silver Badger robot, with the same goals as for the Go2 but with additional cost terms regarding the energy consumption and the minimum required height of the calves, to encourage more *natural* looking robot posture. In both experiments, we set the horizon $H = 16$, number of rollouts $N = 256$, $dt = 20$ ms, temperature $\lambda = 0.05$, horizon and trajectory diffusion factors equal to 0.9 and 0.5,

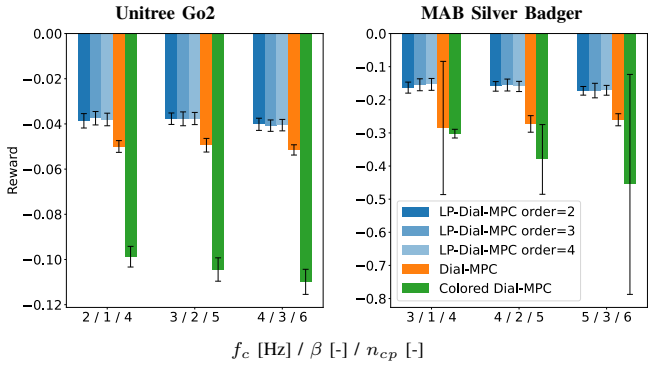

Fig. 7. Comparison of the Dial-MPC [15] with the proposed low-pass filtering and the baseline colored noise [14] perturbations distribution in the task of simulated quadruped locomotion.

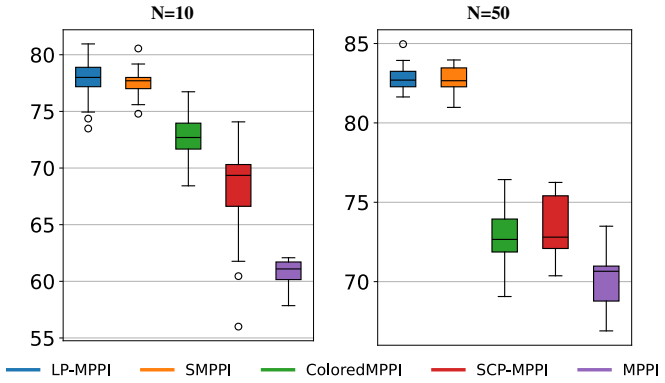

Fig. 8. Distances [m] covered by the F1TENTH car in 30-second time trials using different MPPI algorithm variants. The proposed LP-MPPI achieves the highest median distances both for $N = 10$ and $N = 50$ rollouts.

respectively, and the number of diffusion steps equal to 2.

In this experiment, to highlight the robustness of the proposed approach to the choice of its parameters, we do not perform the search with Optuna but instead report the performance for several intuitive parameter sets, i.e. cutoff frequency $f_c \in \{2, 3, 4\}$ for Unitree and $f_c \in \{3, 4, 5\}$ for MAB robot, and orders $o_{\text{LPF}} \in \{2, 3, 4\}$. We compared their performance with the two above-mentioned baselines, for which we found the sets of the best parameters using Optuna. The results of this experiment can be found in Fig. 7. The proposed low-pass filtering approach implemented into the Dial-MPC framework consistently outperforms the default Dial-MPC by 24% and 41% for the Unitree Go2 and MAB Silver Badger robots, respectively. We attribute these improvements to the more fine-grained control (higher number of decision variables) of the LP Dial-MPC and its ability to directly shape the frequency spectrum of the sampling distribution. In turn, the spectrum shaping capabilities of the colored noise are relatively limited and bias only the lowest frequencies, which results in significantly worse performance (about two times lower rewards than ours).

C. Real-world F1TENTH racing

In all previous experiments, we assumed that the models of the controlled systems are perfectly known and are used by the MPPI to search for the best control trajectories. In turn, in this task, we would like to use an analytical model of the F1TENTH car (dynamic single-track model [29] with MF6.1 tire model [30]) and evaluate it under the real-world racing conditions (see Fig. 4). The goal of this task is to cover the highest possible distance around the track centerline in 30 s, on the $14.2\,\text{m}$ long $1\,\text{m}$ wide oval racetrack. We defined the cost function by

$$c = -v_f + 100\log(1 + \exp(-100(T_w/2 - n))) + 100\max(\alpha - 0.3, 0) + 2(\theta - T_\theta)^2,$$

where v_f is the velocity along the centerline, n is the distance to the centerline, T_w is the track width, α is the slip angle, θ and T_θ are the orientations of the vehicle and the track centerline. The inclusion of the last two terms encourages searching for trajectories that are not drifting and delicately regularizes the trajectories to avoid large orientation deviations. We set the horizon $H = 30$, $dt = 50$ ms, control frequency to 30 Hz, and evaluated the algorithms for both $N = 10$ and $N = 50$ rollouts. We have chosen the number of rollouts, the horizon, and the control frequency to meet real-time requirements with a single core of Intel Core i5-12500H CPU, while achieving reasonable driving performance. The parameters of all methods were chosen in simulation using Optuna [28] with 50 trials.

In Fig. 8, we present the distances covered by the proposed LP-MPPI and the other considered MPPI variants in 15 runs of 30 s each. One can see that in both considered setups, the proposed LP-MPPI approach significantly outperformed all baselines except SMPPI, which performed very close to the LP-MPPI for $N = 50$ and a bit worse for $N = 10$. Note that in the racing scenarios, even the $30\,\text{cm}$ of difference gained every 30 s of the race (the difference between medians of LP-MPPI and SMPPI) may be considered a notable gap.

D. Computational overhead

An important aspect of every control method is its computational efficiency. Therefore, we evaluate how much computational overhead the proposed method introduces relative to the nominal MPPI and how it relates to the other considered baselines. To do so, we run all methods for 10 episodes in Ant-v5 (MuJoCo model, $H = 15, N = 100$) and F1TENTH (compiled analytical model, $H = 30, N = 10$) environments, using a single core of the Intel Core i5-12500H CPU. We compute the median of the control computation time and relate it to the one obtained by the MPPI. In Table II, we present the results of this experiment. One can see that all baselines introduce some notable computational overhead in the F1TENTH environment, since the compiled analytical model, which is responsible for most of the computations, is very fast. Note that the proposed method introduces the second smallest overhead of 2.4%. In turn, in the case of a relatively heavy dynamics model, e.g., Ant-v5 environment, we observe some counterintuitive results, like the decrease in the compute time for LP-MPPI and ColoredMPPI. We suppose that this may be caused by the variability in the timings, due to the use of a standard OS instead of a real-time

TABLE II

COMPUTATIONAL OVERHEAD RELATIVE TO MPPI

Environment	**LP-MPPI**	ColoredMPPI	SMPPI	SCP-MPPI
F1TENTH	+2.41%	+5.12%	+3.89%	**+1.31%**
Ant-v5	**-1.62%**	-0.67%	+3.04%	+1.82%

one, or be an effect of filtering out the higher frequencies from the control signal, which may simplify the underlying physics simulation. In summary, a computationally intense dynamics evaluation causes the overhead introduced by the proposed method to be negligible.

V. CONCLUSIONS

In this work, we introduced Low-Pass Model Predictive Path Integral Control (LP-MPPI), a novel and easy-to-implement enhancement to MPPI that incorporates low-pass filtering into the sampling process. By directly shaping the frequency spectrum of control trajectory perturbations, LP-MPPI eliminates harmful high-frequency noise and improves the efficiency of searching for optimal control trajectories. Unlike existing smoothing techniques or colored noise sampling, our approach offers intuitive fine-grained control over the optimal control search in the frequency domain, making it highly adaptable to various robotic systems.

Through extensive simulation and real-world experiments, we demonstrated the superiority of LP-MPPI over state-of-the-art MPPI-based methods in a variety of tasks, including simulated legged locomotion and real-world F1TENTH autonomous racing. Our results show that LP-MPPI consistently outperforms state-of-the-art methods by 10% in Gymnasium environments, 32% in simulated quadruped locomotion, and by $0.115\,\mathrm{s}$ in a $30\,\mathrm{s}$ long F1TENTH autonomous time trial. In addition, it significantly reduces the chattering of the control signal, leading to smoother and more reliable actuation. Moreover, LP-MPPI maintains computational efficiency, introducing only a negligible overhead compared to standard MPPI, making it practical for real-time applications.

To sum up, LP-MPPI represents a simple yet powerful modification to MPPI, making it an attractive option for real-time robotic control tasks requiring both high-performance trajectory optimization and smooth, actuator-friendly control signals. Future work will explore adaptive filtering techniques to dynamically adjust the sampling distribution based on task demands and further integrate LP-MPPI with learning-based sampling strategies for improved adaptability.

REFERENCES

[1] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger, "Learning-Based Model Predictive Control for Autonomous Racing," *IEEE Rob. and Autom. Lett.*, vol. 4, no. 4, pp. 3363–3370, 2019.

[2] G. Williams *et al.*, "Aggressive driving with model predictive path integral control," in *2016 IEEE Int. Conf. on Rob. and Autom. (ICRA)*, 2016, pp. 1433–1440.

[3] M. Krinner *et al.*, "MPCC++: Model Predictive Contouring Control for Time-Optimal Flight with Safety Constraints," in *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024.

[4] J.-R. Chiu *et al.*, "A collision-free mpc for whole-body dynamic locomotion and manipulation," in *2022 Int. Conf. on Rob. and Autom. (ICRA)*, 2022, pp. 4686–4693.

[5] J. Frey, A. Nurkanovic, and M. Diehl, "Advanced-Step Real-time Iterations with Four Levels – New Error Bounds and Fast Implementation in acados," *IEEE Control Systems Lett.*, vol. 8, pp. 1703–1708, 2024.

[6] R. Verschueren *et al.*, "acados – a modular open-source framework for fast embedded optimal control," *Math. Program. Comput.*, 2021.

[7] M. Minařík, R. Pěnička, V. Vonásek, and M. Saska, "Model predictive path integral control for agile unmanned aerial vehicles," in *2024 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2024, pp. 13 144–13 151.

[8] C. Pinneri *et al.*, "Sample-efficient cross-entropy method for real-time planning," in *Conf. on Robot Learning 2020*, 2020.

[9] M. Bhardwaj *et al.*, "STORM: An integrated framework for fast joint-space model-predictive control for reactive manipulation," 2021.

[10] R. Rubinstein, "The cross-entropy method for combinatorial and continuous optimization," *Methodology And Computing In Applied Probability*, vol. 1, no. 2, pp. 127–190, Sep 1999.

[11] X. Zhang, L. Wan, and X. Ran, "Research progress on the chatter stability in machining systems," *The Int. Journal of Advanced Manufacturing Technology*, vol. 131, no. 1, pp. 29–62, Mar 2024.

[12] T. Miura, N. Akai, K. Honda, and S. Hara, "Spline-interpolated model predictive path integral control with stein variational inference for reactive navigation," in *2024 IEEE Int. Conf. on Rob. and Autom. (ICRA)*, 2024, pp. 13 171–13 177.

[13] T. Kim *et al.*, "Smooth model predictive path integral control without smoothing," *IEEE Rob. and Autom. Lett.*, vol. 7, no. 4, pp. 10 406–10 413, 2022.

[14] B. Vlahov *et al.*, "Low frequency sampling in model predictive path integral control," *IEEE Rob. and Autom. Lett.*, vol. 9, no. 5, pp. 4543–4550, 2024.

[15] H. Xue *et al.*, "Full-order sampling-based mpc for torque-level locomotion control via diffusion-style annealing," in *2025 IEEE Int. Conf. on Rob. and Autom. (ICRA)*, 2025, pp. 4974–4981.

[16] M. Lee and D. Lee, "Time-correlated model predictive path integral: Smooth action generation for sampling-based control," in *2025 IEEE Int. Conf. on Rob. and Autom. (ICRA)*, 2025, pp. 14 490–14 496.

[17] T. Howell *et al.*, "Predictive sampling: Real-time behaviour synthesis with mujoco," arXiv: 2212.00541, 2022.

[18] P. Kicki *et al.*, "Bridging the gap between learning-to-plan, motion primitives and safe reinforcement learning," in *Proc. of The 8th Conf. on Robot Learning*, vol. 270. PMLR, 2025, pp. 2655–2678.

[19] I. S. Mohamed, K. Yin, and L. Liu, "Autonomous navigation of agvs in unknown cluttered environments: Log-mppi control strategy," *IEEE Rob. and Aut. Lett.*, vol. 7, no. 4, pp. 10 240–10 247, 2022.

[20] Z. Wang *et al.*, "Variational Inference MPC using Tsallis Divergence," in *Proceedings of Robotics: Science and Systems*, Virtual, July 2021.

[21] A. Lambert *et al.*, "Stein variational model predictive control," in *Proceedings of the 2020 Conf. on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 155. PMLR, 16–18 Nov 2021, pp. 1278–1297.

[22] T. Power and D. Berenson, "Variational Inference MPC using Normalizing Flows and Out-of-Distribution Projection," in *Proceedings of Robotics: Science and Systems*, New York City, NY, USA, June 2022.

[23] J. Sacks and B. Boots, "Learning sampling distributions for model predictive control," in *Proceedings of The 6th Conf. on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 205. PMLR, 14–18 Dec 2023, pp. 1733–1742.

[24] K. J. Keesman, *System Identification*. London, England: Springer, May 2011.

[25] A. Raffin, "Rl baselines3 zoo," https://github.com/DLR-RM/rl-baselines3-zoo, 2020.

[26] S. Butterworth, "On the Theory of Filter Amplifiers," *Experimental Wireless & the Wireless Engineer*, vol. 7, pp. 536–541, Oct. 1930.

[27] M. Towers *et al.*, "Gymnasium: A standard interface for reinforcement learning environments," *arXiv preprint arXiv:2407.17032*, 2024.

[28] T. Akiba *et al.*, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, 2019.

[29] J. Węgrzynowski, G. Czechmanowski, P. Kicki, and K. Walas, "Learning dynamics models for velocity estimation in autonomous racing," in *2024 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2024, pp. 972–979.

[30] A. J. S. I. J.M. Besselink and H. B. Pacejka, "An improved Magic Formula/Swift tyre model that can handle inflation pressure changes," *Vehicle System Dynamics*, vol. 48, no. sup1, pp. 337–352, 2010.