Singulating an item from a pallet layer: Dual-arm manipulation with minimalistic end effectors by means of sampling-based MPC

Anonymous Authors*

***Abstract*— This work addresses the challenge of picking an item from a orderly-arranged layer of objects by means of dual-arm manipulation with minimalistic end-effectors. The task is inspired by manual depalletization, a common material-handling process in logistics. Successful execution of the task requires multiple sequential motions to isolate an object before both arms can hold it firmly. Motivated by the recent availability of parallel physics simulators, we explore the feasibility of sampling-based Model Predictive Control (MPC) to solve the combined motion planning and control problem online, as a complementary approach to offline-computed reinforcement learning policies. We propose a task-specific cost formulation and combine MPPI temperature adaptation with control input spline parametrization, to retain high-success rate with limited optimization parameters and thus reduced computational burden. We benchmark the effectiveness of the approach by means of a numerical study, against naive baseline implementations.**

I. Introduction

Dexterous manipulation is still an open problem for robotics. Within this context, in this work we focus on the challenging task of extracting a single item from a physically tight arrangement of objects by means of two arms provided with minimalist soft-pad end effectors. More specifically, we consider the situation illustrated in Figure 1, where an item must be removed from a layer of boxes by means of pushing, tilting, and holding motions. This object arrangement is illustrative of the type of manual work that a warehouse worker needs to perform while operating at a modern manual depalletizing station. The minimalistic soft pad end effectors can be though as being the palms of the operator. We are drawn by the challenge of extracting a corner box from a pallet layer. Such task requires dexterous intermediate maneuvers because the arms cannot immediately reach the coordinated lifting configuration where the end effectors find themselves on two opposite sides of the item.

The key difficulty in planning this type of robot motions is the need for making or braking contact multiple times until task completion. When employing optimization-based planning, the discontinuous and stiff interaction dynamics yield nonconvex, rapidly changing cost landscapes at contact transitions. Furthermore, in contact-rich tasks it is difficult to predict when or how many contacts will occur, and thus, planning for each contact mode is intractable. Reinforcement Learning (RL) can overcome such difficulties in contact-rich manipulation tasks [1]–[3], such as pick-and-place, by smoothing out the objective via the random sampling nature of exploration [4]. Picking up on this interpretation and driven by advancements in online parallel physics simulation [5]–[7], sampling-based Model Predictive Control (MPC) has recently emerged as a compelling alternative to or, better, a potentially valuable ally of RL [8]–[15], promising online adaptability to environment and task variations.

Sampling-based MPC has been applied in contact-rich tasks in [8]–[10], [12]. However, studies and applications that are related to dual-arm manipulation in the presence of multiple items in clutter or, as in this paper, in a tight object arrangement do not exist yet. Moreover, in most of these works, the objective function is smoothed through sampling with a fixed sampling distribution, leaving the algorithm prone to over-smoothing, which restricts convergence, or under-smoothing, which hinders coverage.

In this paper, we show how sampling-based MPC can address contact-rich dual-arm item picking from an orderly-arranged layer of objects and utilize a heuristic rule which essentially adapts the sampling distribution evolutionary. To the best of our knowledge, this is the first work:

1) addressing online motion synthesis for dual-arm item picking in a tight object arrangement, and
2) investigating the benefits of evolutionary temperature adaptation in MPPI in combination with a spline reparameterization of the control sequence to reduce computational cost by reducing the search space dimension, preserving high-success rate.

We evaluate our approach in numerical simulation for a depalletization use case and show its effectiveness both in quantitative and qualitative terms. We use the MuJoCo physics engine [5] for forward simulation.

Our work is inspired by Model Predictive Path Integral control (MPPI) with the temperature adaptation rule in [10], which outperforms standard MPPI and Cross Entropy Method (CEM) for the task at hand. Moreover, we combine it with the spline reparameterization of the control sequence [11] and optimize over time-indexed spline knots.

The paper is organized as follows. Section II presents the proposed approach covering the developed cost terms and the motion planner. Section III provides quantitative and qualitative empirical evaluation in numerical simulations, followed by a discussion on key findings, limitations, and directions for future work. Finally, Section IV summarizes our contributions.

II. Proposed Approach

This section presents the main components of our approach: 1) the task we tackle and the system we use, 2) a high-level description of the planner, and 3) the cost terms

*Manuscript under double blind review.

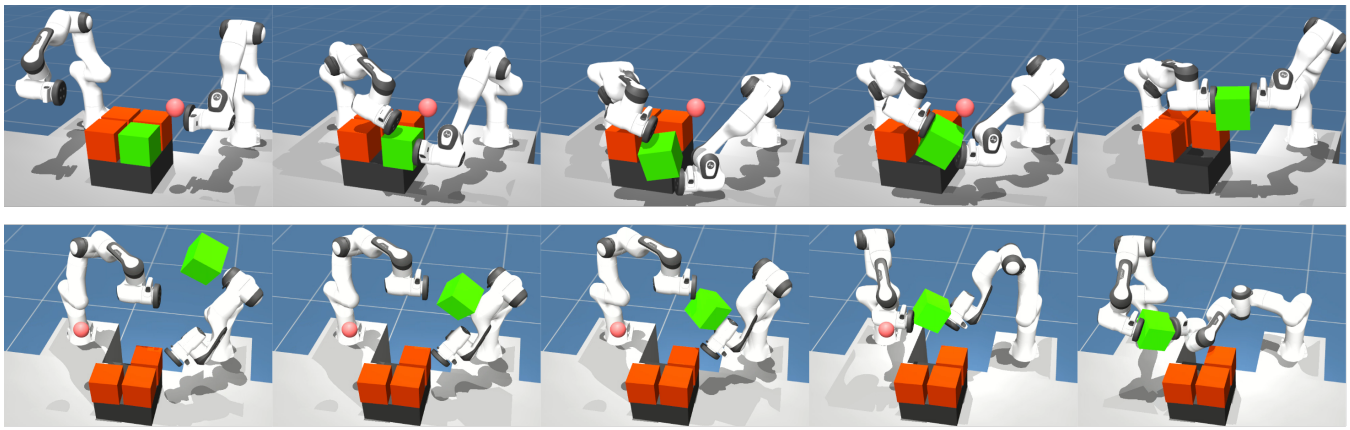

Fig. 1. Collection of frames that depict the generated motions for removing a box from a full-grid of boxes without using any other parts of the arm except the soft pads attached at the end effectors (top row) and for grabbing a box suddenly released in mid-air by leveraging the arms' whole surface (bottom row).

used for sampling-based MPC in the underlying task. The implementation code is available at [to be provided upon acceptance].

A. Dual-arm picking from a cluster of objects

The task under consideration involves picking an object, which we call the focus object, from a physically tight arrangement of objects, leveraging the dexterity of a dual-arm system. For instance, the presence of two arms allows one arm to push or tilt the box while the other arm supports it before a grasp can be achieved.

We employ a dual-arm system with flat pads attached to the end-effectors. We use flat pads instead of more sophisticated end-effector tools since they are sufficient, durable, and cost-effective. Moreover, we consider them to be made out of silicone to provide grasping friction and dampen impacts, improving the grasping capabilities. A physical version of a similar setup was used to demonstrate impact-aware manipulation of logistics parcels [16].

B. Motion planner

We generate motion plans for each end-effector utilizing sampling-based MPC. This section presents an overview of sampling-based MPC, then describes how we model the task and introduces different algorithms.

Sampling-based MPC generates N control sequences in each planning step by injecting noise into an initial guess of the optimal control sequence. A common choice for noise injection is a zero-mean multivariate Gaussian with covariance matrix $\Sigma = \text{diag}(\sigma)$, where $\sigma \in \mathbb{R}$. Each control sequence U is simulated to compute its state trajectory, yielding N state trajectories, called rollouts. Given a cost function $c(x)$ for a state x, we can compute the cumulative cost of the i-th rollout S_i. Then, the new guess is the weighted average of the control sequences $\sum_{i=0}^{N-1} w_i \, U_i$, where w_i is a weight based on S_i. This process is repeated until the task is completed.

In our approach, the planner operates in the Cartesian space, producing a sequence of 6D poses for the two end-effectors. The pose sequence is then filtered by a low-pass filter and then fed to a Cartesian-space compliant controller. Inspired by [11], [14], we parameterize the sequence as time-indexed spline knots and optimize over the value at the knots instead of parameterizing every action value in the discrete-time moving horizon. The moving horizon corresponds to a time length T and a timestep dt. The horizon's timestep is equivalent to the physics engine's integration timestep, and the total number of discrete-time points in the horizon is $K = T/dt$. We call the sequence of time-indexed spline knots a *plan*. The number of spline knots is denoted as N_P and the dimension of each knot as d. The spline-knots reparameterization reduces the search space, enabling online optimization in high-dimensional settings. We tune N_p, T, and dt empirically (cf. Section III-A).

We compare several sampling-based MPC algorithms that differ in the noise injection and weight computation steps. **Predictive Sampling (PS)** [11] samples noise ε^i, where i is the rollout, from a Gaussian Distribution $\mathcal{N}(0,\Sigma)$ with $\Sigma \in \mathbb{R}^{N_P \; d \times N_P \; d}$, $\Sigma = \text{diag}(\sigma)$, and fixed σ. It assigns a weight of 1 to the best-performing rollout (minimum cost) and 0 to every other. Similarly, **Model Predictive Path Integral control (MPPI)** [17] uses $\varepsilon^i \sim \mathcal{N}(0,\Sigma)$ with $\Sigma = \text{diag}(\sigma)$, however, the weights are computed using importance sampling. In MPPI, the temperature parameter λ controls how the weights concentrate or spread over high-performing rollouts. To dynamically adjust the temperature, we employ the heuristic introduced in [10] which aims to keep the sum of the normalized cumulative costs within a predefined range $[\eta_{min}, \eta_{max}]$. In the **Cross-Entropy Method (CEM)** [18], noise injection starts with $\varepsilon^i \sim \mathcal{N}(0,\Sigma)$ with $\Sigma = \text{diag}(\sigma)$. Then, after each planning iteration, Σ is fit to the covariance of the N_e best-performing rollouts. We choose to fit only the diagonal components of the covariance matrix, with each component corresponding to a coordinate of a spline knot, since no correlation between the decision variables is expected, and finding just the variance is computationally cheaper. The N_e best performing rollouts are called elites

and N_e is predefined as a proportion of the total amount of rollouts. In CEM, elite rollouts get a weight equal to $1/N_e$ while all the other rollouts have zero weights.

In all the aforementioned algorithms, there is a tradeoff between coverage and convergence, which is captured by the noise magnitude. Larger magnitudes can help in getting out of local minima, but can also make the solver miss the optimum and vice versa. Inspired by [11], we mitigate this issue by sampling from a mixture of two Gaussians $\varepsilon^i \sim (1-p_e)\ \mathcal{N}(0,\Sigma_1) + p_e\ \mathcal{N}(0,\Sigma_2)$, with $\Sigma_1, \Sigma_2 \in \mathbb{R}^{N_P \times d}$ and $\Sigma_1 = \text{diag}(\sigma_1)$, $\Sigma_2 = \text{diag}(\sigma_2)$, where $\sigma_2 > \sigma_1$. For CEM, Σ_1 is fit to the elite rollouts while Σ_2 is left untouched. This mixture sacrifices a small portion p_e of the total amount of rollouts for exploration using larger perturbations induced by σ_2 while using the smaller σ_1 for the rest of the rollouts.

C. Cost terms

The task of picking a single item from a cluster of objects is encoded by defining a cost function, comprising multiple cost terms. To provide intuition on why and how these cost terms were defined, we provide a 2D illustration of various relevant geometric entities in Fig. 2.

To anticipate contacts, we introduce virtual semi-spheres at the centers of the flat pads (represented as semi-circles in Fig. 2). These semi-spheres exist only in the simulation environment used for calculating rollouts and serve as "detectors" of potentially incoming contacts. Such virtual contacts do not generate forces, and only their contact normal vectors are used. In MuJoCo, contact normal vectors between two colliding geometries are calculated using minimum depth computation algorithms[1].

The cost terms are designed to be independent of the specific robot arms, the object's size, or variations of the task, such as different starting and goal positions, as long as the goal remains to pick an object with a dual-arm system. Apart from the distance-related cost terms (e.g., distance to goal), the terms range roughly between 0 and 1, with higher values indicating less desirable states. Normalizing their values makes tuning the corresponding weights easier (cf. Section III-A). The terms were inspired by [11], with the ones corresponding to dual-arm coordination being novel. A summary for all cost terms is given below with an illustration of the effect of the opposite-contact and collision-surface-alignment terms in Fig. 3:

- **Reach** - Encourage the arms to approach the focus box. They are defined separately for the left and the right arm.
- **Bring** - Penalizes the focus box to be away from the goal position, indirectly encouraging the arms to move it to the desired location.
- **Opposite-contact** - If both arms collide with the focus box, it motivates opposite collision surfaces.
- **Collision-surface-alignment** - If both pads contact the focus box, encourage them to face it.

[1] https://mujoco.readthedocs.io/en/stable/computation/#convex-collisions

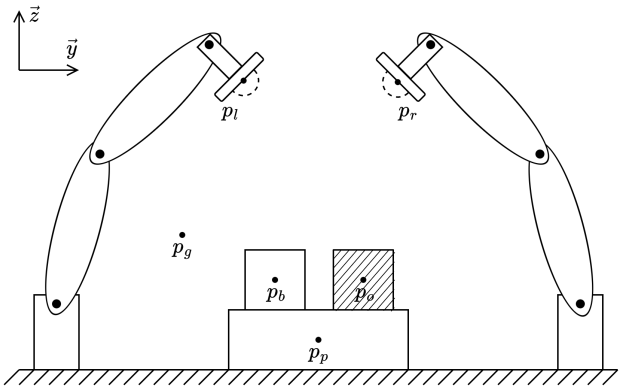

Fig. 2. Planar illustration of the setup, where $\mathbf{p}_p$ is the pallet position, $\mathbf{p}_o$ an obstacle box position, $\mathbf{p}_b$ the focus box position, $\mathbf{p}_g$ the goal position, and $\mathbf{p}_l$ and $\mathbf{p}_r$ the flat pads positions. The dashed semi-circles around the flat pads are the collision detection range, enlarged in the figure for ease of illustration. The y and z axes represent the inertial world frame.

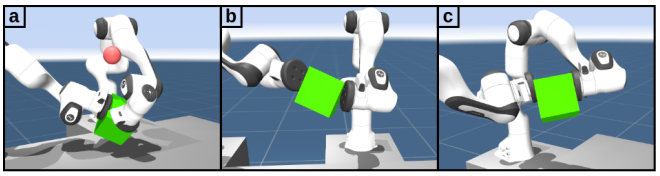

Fig. 3. Snapshots from a simulation: (a) c_{opposite} and c_{align} are not used; (b) c_{opposite} is used and c_{align} is not; (c) both c_{opposite} and c_{align} are used.

- **Anti-parallel** - Motivates the end-effectors to have opposite directions even when no collisions occur.
- **Y-difference** - The y-difference cost term penalizes one arm going over the other.
- **Careful** - Penalizes unintentional contacts.
- **Pallet** - Prevents displacement of the pallet.

III. Numerical Simulations and Discussion

This section presents the results of experiments in numerical simulation that evaluate the effectiveness of our approach in picking objects from a cluster. The task involves grabbing the focus box from a 2D cluster of 2 by 2 boxes placed on top of a mini-pallet (cf. Fig. 1). In this scenario, the motions needed for extracting the focus box are representative of the ones needed for performing dual-arm depalletization. The boxes are modeled as $14\,\text{cm} \times 14\,\text{cm} \times 14\,\text{cm}$ cubes and the pallet as a $30\,\text{cm} \times 30\,\text{cm} \times 14\,\text{cm}$ cuboid. The task is defined in such a way that it requires dexterous coordinated motions for object singulation instead of motions that push the other boxes away to free space around the focus box (e.g., [19]). As supplement, we provide a series of videos demonstrating the generated motions (see submission attachments).

A. Empirical Tuning

We tune the cost terms weights and the planner's hyperparameters using an interactive Graphical User Interface (GUI), where hyperparameters can be adjusted while observing the system's behavior. This is an inherent feature of MuJoCo MPC, which allows a human operator to empirically tune

these parameters. The process can be automated and even optimized by using hyperparameter optimization methods. However, running such methods can be time-consuming, and we are interested in a simple and easy-to-adapt solution.

B. Quantitative analysis

To evaluate our approach quantitatively, we consider 4 scenarios. In each scenario, the focus box in the 2×2 pallet is varied and the remaining boxes are obstacles. For each scenario, we run an equal number of episodes. The episode ends successfully if the focus box is moved to the goal within 30 seconds and no box falls. The metrics we consider are: the success rate (S), the average time for reaching the goal (AR), the average time the system managed to hold the box at the goal (AH), the fail rate due to a fall of the focus box (F1), the fail rate due to a fall of another box (F2), and the timeout rate (T). These metrics capture both whether the system completes the task and how well it performs.

We conduct experiments using different planners (cf. Section II-B) and investigate the effects of the amount of rollouts and the planner's preference towards the better performing rollouts. See Tables I and II for CEM and MPPI results, respectively. A systematic analysis of PS and MPPI without the λ adaptation rule is not included, as they underperformed during qualitative testing with respect to the other two.

From the results, we find that, for both methods, more rollouts (N) do not necessarily yield better performance. Fewer rollouts imply lower computational time and, due to the asynchronous nature of the approach, a higher replanning frequency, which appears to result in higher success rates. On the other hand, reaching the goal alone is not enough. The system should be able to hold the box steadily, and typically, fewer rollouts result in lower AH times and vice versa.

MPPI with temperature adaptation seems to handle better than CEM the tradeoff between jerky motions, which enable extracting the box (coverage), and stable motions, which enable holding the box (convergence). We get an overall best for both success rate and AH time when using 39 rollouts with MPPI and $[\eta_{min}, \eta_{max}] = [5, 10]$. We conjecture that this happens due to the evolutionary adaptation of the temperature, which shifts importance either towards or away from higher cost rollouts to keep η inside its range.

Overall, with both MPPI and CEM we get a success rate of about 80%, which is surprising given the online nature of the approach, with no biasing from previous experience.

C. Qualitative analysis

To illustrate which motions the planner generates, 5 frames are extracted from an episode: see Fig. 1 (top). In the beginning of the episode, the left arm, that is further away, reaches the focus box from above and pushes it towards the edge of the pallet. The right arm holds the box while it is pushed and as soon as opposite collisions are detected, the right arms moves towards the side of the box that faces the ground (opposite from the top side where the left arm is). Finally, the arms grab the box and move it to target.

TABLE I

CEM PERFORMANCE FOR DIFFERENT AMOUNTS OF ROLLOUTS (**N**). THE VALUES BEFORE AND AFTER THE VERTICAL BAR CORRESPOND RESPECTIVELY TO USING 10% AND 20% OF THE TOTAL AMOUNT OF ROLLOUTS AS ELITES.

N	**S**(%)↑	**AH**(s)↑	**AR**(s)↓	**F1**(%)↓	**F2**(%)↓	**T**(%)↓
13	59|66	3.8|6.6	8.4|8.7	17|14	22|18	**2**|**2**
26	**78**|62	11.4|13.7	8.6|9.1	**7**|10	11|6	4|22
39	60|70	16.1|16.5	8.7|8.8	14|8	8|5	18|17
52	62|66	16.5|**19.3**	7.5|**7.1**	15|**7**	14|4	9|23
65	71|60	15.2|16.2	8.4|11.2	10|**7**	4|3	15|30
78	59|57	17.6|18.8	8.8|8.7	16|8	12|**2**	13|33

TABLE II

MPPI PERFORMANCE FOR DIFFERENT NUMBER OF ROLLOUTS (**N**). THE VALUES BEFORE AND AFTER THE VERTICAL BAR CORRESPOND RESPECTIVELY TO USING $[5, 10]$ AND $[9, 20]$ AS $[\eta_{min}, \eta_{max}]$.

N	**S**(%)↑	**AH**(s)↑	**AR**(s)↓	**F1**(%)↓	**F2**(%)↓	**T**(%)↓
13	55|0	0.7|0.0	13.2|inf	21|28	13|62	11|10
26	81|56	13.2|0.9	5.6|16.4	**4**|13	6|**1**	9|30
39	**82**|71	**20.4**|10.9	**4.8**|8.3	**4**|5	7|6	**7**|18
52	74|71	19.4|16.1	6.7|7.0	7|9	9|7	10|13
65	56|64	18.6|18.6	6.7|8.2	18|12	11|3	15|21
78	71|69	18.8|18.4	7.3|7.4	13|14	9|6	7|11

As an extension, we modify the careful cost term in such a way that allows contacts of all the parts of the arms with the focus box. Then, we manually position the focus box, using the interactive GUI, well above the robot arms and hold it for a while (cf. Fig. 1 (bottom)). At some point, we release it in mid-air. The planner initially leverages the right arm's elbow and then the rest of the arm's surface to guide the focus box between the two end-effectors. Eventually, the arms grab the box and move it to the goal. This example suggests that the motion planner could leverage the use of the arms' whole surface for manipulation.

IV. CONCLUSION

This work presents an online motion synthesis approach for dual-arm dexterous picking of objects compacted in a cluster. To accomplish the task, we introduce several cost terms since the standard and more abstract ones, which only minimize the arms' distance from the object and the object's distance from the goal, were not enough. Given these cost terms, we utilize sampling-based MPC to generate motion plans for a novel dexterous depalletization use case by means of numerical simulations, where coordinated motions of the two arms are needed to first extract and eventually move a box from a pallet's full layer. We find combining a spline-knots reparameterization with an adaptive temperature variation of MPPI to be the most effective method of the ones we test, achieving surprisingly good results for a purely online method. Lastly, we manually interact with the simulated environment and show in qualitative terms that the same planner can potentially leverage the arms' whole surface for manipulating objects without any modifications to the approach, besides removing the object-arm contact penalty.

REFERENCES

[1] H. Qi, B. Yi, S. Suresh, M. Lambeta, Y. Ma, R. Calandra, and J. Malik, "General In-hand Object Rotation with Vision and Touch," in *Proc. 7th Conference on Robot Learning (CoRL)*, 2023, pp. 2549–2564.

[2] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware," in *Proc. Robotics: Science and Systems (RSS)*, vol. 19, 2023.

[3] Z. Luo, J. Cao, S. Christen, A. Winkler, K. Kitani, and W. Xu, "Omnigrasp: Simulated Humanoid Grasping on Diverse Objects," in *Proc. 39th Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[4] H. J. Suh, M. Simchowitz, K. Zhang, and R. Tedrake, "Do Differentiable Simulators Give Better Policy Gradients?" in *Proc. 39th International Conference on Machine Learning (ICML)*, 2022, pp. 20 668–20 696.

[5] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 5026–5033.

[6] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac Gym: High Performance GPU-Based Physics Simulation For Robot Learning," 2021. [Online]. Available: http://arxiv.org/abs/2108.10470

[7] S. Tao, F. Xiang, A. Shukla, Y. Qin, X. Hinrichsen, X. Yuan, C. Bao, X. Lin, Y. Liu, T.-k. Chan, Y. Gao, X. Li, T. Mu, N. Xiao, A. Gurha, Z. Huang, R. Calandra, R. Chen, S. Luo, and H. Su, "ManiSkill3: GPU Parallelized Robotics Simulation and Rendering for Generalizable Embodied AI," 2024. [Online]. Available: http://arxiv.org/abs/2410.00425

[8] I. Hurova, A. Dan, K. Kruusamäe, and A. K. Singh, "Sampling-based optimization with parallelized physics simulator for bimanual manipulation," 2025. [Online]. Available: http://arxiv.org/abs/2511.21264

[9] D. Russell, Z. Xu, M. A. Roa, and M. Dogar, "Pack it in: Packing into partially filled containers through contact," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2026.

[10] C. Pezzato, C. Salmi, E. Trevisan, M. Spahn, J. Alonso-Mora, and C. Hernández Corbato, "Sampling-Based Model Predictive Control Leveraging Parallelizable Physics Simulations," *IEEE Robotics and Automation Letters (RA-L)*, vol. 10, no. 3, pp. 2750–2757, 2025.

[11] T. Howell, N. Gileadi, S. Tunyasuvunakool, K. Zakka, T. Erez, and Y. Tassa, "Predictive Sampling: Real-time Behaviour Synthesis with MuJoCo," 2022. [Online]. Available: http://arxiv.org/abs/2212.00541

[12] A. H. Li, P. Culbertson, V. Kurtz, and A. D. Ames, "DROP: Dexterous Reorientation via Online Planning," in *Proc. International Conference on Robotics and Automation (ICRA)*, 2025.

[13] A. Hess, A. M. Kübler, B. Forrai, M. Dogar, and R. K. Katzschmann, "Sampling-Based Model Predictive Control for Dexterous Manipulation on a Biomimetic Tendon-Driven Hand," in *Proc. International Conference on Intelligent Robots and Systems (IROS)*, 2025.

[14] J. Alvarez-Padilla, J. Z. Zhang, S. Kwok, J. M. Dolan, and Z. Manchester, "Real-Time Whole-Body Control of Legged Robots with Model-Predictive Path Integral Control," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[15] H. Xue, C. Pan, Z. Yi, G. Qu, and G. Shi, "Full-Order Sampling-Based MPC for Torque-Level Locomotion Control via Diffusion-Style Annealing," in *Proc. International Conference on Robotics and Automation (ICRA)*, 2025.

[16] J. van Steen, G. van den Brandt, N. van de Wouw, J. Kober, and A. Saccon, "Quadratic Programming-Based Reference Spreading Control for Dual-Arm Robotic Manipulation With Planned Simultaneous Impacts," *IEEE Transactions on Robotics (T-RO)*, vol. 40, pp. 3341–3355, 2024.

[17] 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 (T-RO)*, vol. 34, no. 6, pp. 1603–1622, 2018.

[18] M. Kobilarov, "Cross-entropy motion planning," *The International Journal of Robotics Research (IJRR)*, vol. 31, no. 7, pp. 855–871, 2012.

[19] Y. Wang and H. Kasaei, "Learning Dual-Arm Push and Grasp Synergy in Dense Clutter," *IEEE Robotics and Automation Letters (RA-L)*, vol. 10, no. 5, pp. 5154–5161, 2025.