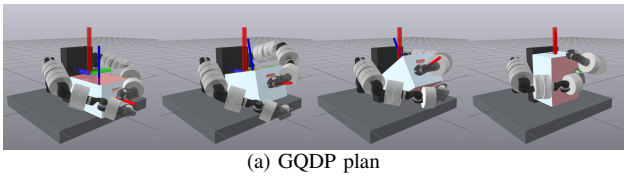
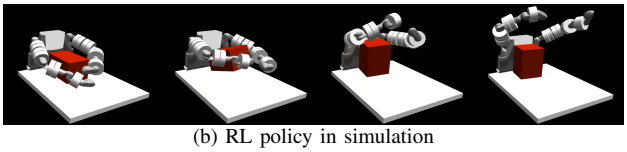
Plan-Guided Reinforcement Learning for Whole-Body Manipulation

Mengchao Zhang[†1], Jose Barreiros[†2], Aykut Özgün Önol[†2]

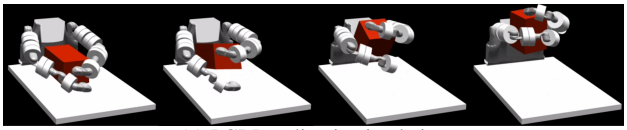
***Abstract*— Synthesizing complex whole-body manipulation behaviors has fundamental challenges due to the rapidly growing combinatorics inherent to contact interaction planning. While model-based methods have shown promising results in solving long-horizon manipulation tasks, they often work under strict assumptions, such as known model parameters, oracular observation of the environment state, and simplified dynamics, resulting in plans that cannot easily transfer to hardware. Learning-based approaches, such as imitation learning (IL) and reinforcement learning (RL), have been shown to be robust when operating over in-distribution states; however, they need heavy human supervision. Specifically, model-free RL requires a tedious reward-shaping process. IL methods, on the other hand, rely on human demonstrations that involve advanced teleoperation methods. In this work, we propose a plan-guided reinforcement learning (PGRL) framework to combine the advantages of model-based planning and reinforcement learning. Our method requires minimal human supervision because it relies on plans generated by model-based planners to guide the exploration in RL. In exchange, RL derives a more robust policy thanks to domain randomization. We test this approach on a whole-body manipulation task on Punyo, an upper-body humanoid robot with compliant, air-filled arm coverings, to pivot and lift a large box. Our preliminary results indicate that the proposed methodology is promising to address challenges that remain difficult for either model- or learning-based strategies alone.**



(a) GQDP plan



(b) RL policy in simulation



(c) PGRL policy in simulation

(d) PGRL policy on hardware

Fig. 1: Snapshots of the GQDP plan (a) simulation (b, c) and hardware (d) policy rollouts to show the style difference between the plan and the RL and PGRL policies, as well as the effective sim-to-real transfer.

I. Introduction

Humans employ a diverse array of strategies to effectively manipulate various objects, including dexterous manipulation, full-body engagement, and interactions with the environment [1, 2]. In the realm of robotics, there has been a longstanding endeavor to replicate and integrate these intricate human behaviors. For instance, consider the scenario delineated in Fig. 1(a), wherein a robot is tasked to move a large box to a desired pose. In this pursuit, the robot might initiate the process by employing both arms to pull the box toward its torso, subsequently utilizing a hand to pivot the box by bracing it against its torso until it attains the desired configuration. However, devising a framework to systematically plan and control the execution of such contact-rich behaviors presents a formidable challenge.

Model-based planners struggle with these hybrid-dynamics problems because contacts lead to stiff, non-smooth numerics with an excessive number of discrete contact modes. Although recent advancements in planning with contacts have exhibited promising outcomes in manipulation planning [3–8], the robust execution of the planned trajectory in hardware remains an unresolved issue. Often, these planners take a considerable amount of time to converge and run offline when trying to discover complex contact sequences. Open-loop execution manifests susceptibility to uncertainties in model parameters and initial pose [4], so achieving robust execution necessitates closing the loop.

Imitation learning (IL) emerges as a promising pathway to tackle this challenge, a prospect bolstered by recent progress in gradient-field learning methods [9–11]. Yet, applying this strategy to whole-body manipulation tasks confronts impediments originating from the limitations of teleoperation methodologies. Predominantly tailored for end-effector tracking, these techniques fall short when tasked with effectively showcasing intricate whole-body maneuvers. Even when resorting to whole-body teleoperation techniques (such as motion-capture-based kinematic retargeting) for generating demonstrations, our empirical observations indicate limitations stem from the absence of comprehensive whole-body haptic feedback [12, 13] available to the teleopera-

[†] Equal contribution.

[1]Department of Mechanical Science and Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, USA 61801. mz17@illinois.edu

[2]Toyota Research Institute, Cambridge, MA, USA, 02139. {jose.barreiros, aykut.onol}@tri.global

*This work was done during Mengchao Zhang's internship at Toyota Research Institute.

This paper has supplementary downloadable material provided by the authors. It includes an MP4 format movie clip [Link], which shows demonstrations of our method.

tor. Furthermore, the substantial volume of demonstrations required to effectively train a proficient imitation learning policy has emerged as another restrictive factor.

Recent advances in reinforcement learning (RL) have yielded remarkable outcomes in dexterous manipulation [14–17] that are difficult to replicate with model-based planning. Notably, these advancements frequently hinge upon the availability of task-specific insights, either in the form of well-defined reward functions or expert guidance. The acquisition of such task-related information, however, can pose difficulties, particularly in the domain of whole-body manipulation.

As a means to streamline the process of reward design, guided RL capitalizes on pre-existing knowledge inferred from data to enhance the efficiency and efficacy of the reinforcement learning process [18]. In particular, example (or demonstration)-guided RL that aims to combine motion imitation with task-based rewarding (aka. standard RL) has been a popular concept in robotics (as well as in animation) to aid exploration by instilling a desired motion style, accelerate learning, and ease reward shaping [19–29].

Peng et al. [30] recently proposed an example-guided RL framework based on adversarial imitation learning, named Adversarial Motion Priors (AMP). This approach does not require designing imitation objectives or motion selection mechanisms, and it can automatically synthesize a policy that completes a desired high-level task given a set of unstructured example motions. Due to its promise to impose motion characteristics on the RL policy without the burden of reward engineering, the AMP idea which was originally proposed for animation, has caught the attention of the robotics community and led to impressive results for quadruped locomotion, e.g., [31–35]. These are compelling examples showing that the AMP approach can generate policies: (i) with a desired style even from infeasible, partial, and scarce demonstrations; and (ii) that can directly transfer to the real world.

However, producing demonstrations through teleoperation for intricate whole-body manipulation tasks remains a challenging endeavor due to relying on human supervision, which hinders the potential of this method to scale to the large set of diverse skills needed for a robot to operate in the real world. As a potential solution, state-of-the-art global-planning-through-contact methods, such as the Global Quasi-Dynamic Planner (GQDP) [3], can synthesize long-horizon behaviors with complex intermittent contact interactions, yet the resulting plans are fragile (even with a perfect model) when executed in an open-loop fashion. Our hypothesis is that AMP can help us convert a rough (and potentially infeasible) plan into a feedback policy that can immediately be deployed on hardware. It is noteworthy that while it is possible to track plans with conventional control methods that often entail estimations for hidden model parameters, learning approaches let us work directly with output feedback.

Imitating plans is not a new idea and has been previously studied, especially for quadruped locomotion, e.g., [36–39]. In the context of manipulation, Wang et al. [40] proposed using a motion and grasp planner for two purposes: (i) to generate demonstrations for off-policy learning and (ii) to supervise an auxiliary-goal, grasp-pose-prediction task. As a result, they obtain closed-loop grasping and human-robot handover behaviors. Huang et al. [41] propose a diffusion-based generative sampling framework that can provide physics-aware goal-oriented plans given a 3D scene. They showcase this approach for dexterous grasp generation and collision-aware arm motion planning using offline-generated plans. However, due to its pure imitation nature, this method requires several high-fidelity plans.

Despite exciting advancements in leveraging planning to guide RL for locomotion problems, this has not yet translated to dexterous manipulation, probably due to the unavailability of reliable planners until recently. In this work, we introduce a simple but effective framework named Plan-Guided Reinforcement Learning (PGRL), which is illustrated in Fig. 2. This framework, while planner agnostic, first uses the GQDP to synthesize quasi-dynamic plans which are not necessarily feasible, given the desired pose of the manipuland. Then, the AMP is used to complete the gaps in the plan and derive a closed-loop policy with similar motion characteristics to the plan. We test this approach to address a complex whole-body manipulation task employing Toyota Research Institute's Punyo robot [42]. Our preliminary findings show that this approach can enable generating policies efficiently even from a single, infeasible example plan. Notably, our method obviates the requirement for human demonstrations and operates in conjunction with a simplified reward function. Facilitated by domain randomization and the inherent robustness stemming from Punyo's passive compliance, the trained policies can be seamlessly transferred to real hardware without necessitating subsequent processing. While we present an instance of this approach using GQDP, we emphasize that this approach is not limited to a certain source for the demonstration and could be used with other types of planners and teleoperation data. Nevertheless, the impact of the demonstration type on the performance is an open question.

II. Method

In this section, we briefly describe the methods we use for the proposed framework. The implementation details can be found in section V-A.

A. Planning Through Contact

We select GQDP [3] as the contact-implicit planner because of its capability for synthesizing long-horizon behaviors with multiple intermittent contacts. This approach assumes quasi-dynamics to reduce the problem into the configuration space and uses a contact smoothing scheme that along with the locally linear model is used to derive a reachability metric. As a result, the planning through contact problem given initial and desired configurations can be solved by a sampling-based planner, in this case, the rapidly exploring random tree (RRT) method [43]. The path generated by RRT is then refined using trajectory optimization to output

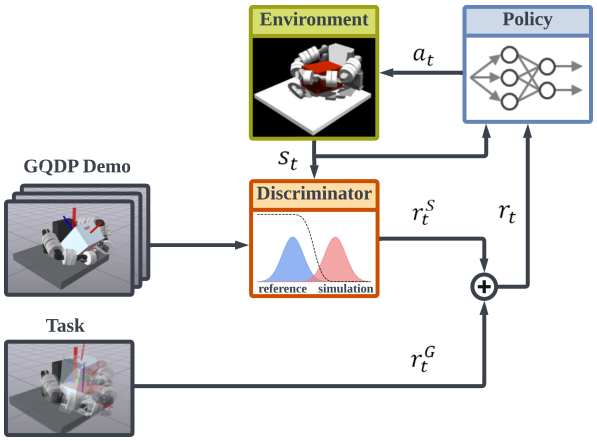


Fig. 2: A flowchart of the proposed framework: the process commences with the acquisition of a reference motion dataset, generated through the utilization of a model-based planner; then the system undertakes the training of a discriminator, designed to acquire proficiency in learning an imitation reward, and a policy, which in turn empowers the robot to replicate the demonstrated motion style while simultaneously accomplishing the intended task.

a trajectory denoted as $\mathcal{T}' = \{(\mathbf{q}_t^a, \mathbf{q}_t^u, \mathbf{a}_t) | t = 0, \cdots, T\}$. Here, $\mathbf{q}_t^a$ and $\mathbf{q}_t^u$ are the configurations of the actuated and unactuated degrees of freedom of the system, and $\mathbf{a}_t$ denotes the robot position commands (or actions) for a joint stiffness controller at each discrete time step t.

It is pertinent to note that the resulting plan may exhibit non-physical behavior because of: (i) the quasi-dynamic assumption breaking when the system moves at non-negligible speeds and (ii) a robot teleport issue when switching contact modes. Due to (ii), the manipuland is not guaranteed to remain stable when the robot transitions from one contact configuration to another. This entails that while the robot follows the planned waypoints $\mathbf{q}_t^a$, the manipuland may not stay at the corresponding $\mathbf{q}_t^u$ for unstable tasks. Consequently, in our approach, we retain only the robot's configuration sequence $\mathcal{T} = \{\mathbf{q}_t^a | t = 0, \cdots, T\}$ as the demonstration.

B. Example-Guided Reinforcement Learning

Given a dataset of reference motions and a task objective defined by a reward function, AMP synthesizes a control policy that enables the agent to achieve the task objective, while utilizing behaviors that resemble the motion style of the dataset. It is important to emphasize that the agent's behaviors are not obligated to precisely replicate individual motions from the dataset. Rather, the agent is encouraged to embody the broader characteristics observed within the collection of reference motions. This renders the GQDP a promising candidate for serving as the demonstrator. The emphasis here lies not on the successful task completion through the plan, but rather on utilizing it as a stylistic guide to steer the robot's behavior and increase the training efficiency and effectiveness.

Our policy is trained through an RL framework, wherein an agent engages with an environment in accordance with a policy denoted as π. At each time step t, the agent perceives the state $\mathbf{s}_t \in \mathcal{S}$ of the system. It then proceeds to sample an action $\mathbf{a}_t \in \mathcal{A}$ from the policy, adhering to the probabilistic distribution $\mathbf{a}_t \sim \pi(\mathbf{a}_t | \mathbf{s}_t)$. Subsequently, the agent runs this chosen action, leading to a new state $\mathbf{s}_{t+1}$, accompanied by a scalar reward $r_t = r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1})$. In accordance with [30], we formulate the reward function as comprising two distinct components: (i) the task reward r^G that quantifies the degree of task accomplishment, and (ii) the style reward r^S that assesses the resemblance between the robot's motion and the reference motions:

$$r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}) = \lambda r^G(\mathbf{s}_{t+1}, \mathbf{a}_t) + (1 - \lambda) r^S(\mathbf{s}_t, \mathbf{s}_{t+1}),$$

where $\lambda \in [0, 1]$ determines the task reward weight with respect to the imitation reward weight.

To minimize reward shaping, we choose to employ a straightforward task-reward function $r^G = d_{trans} + d_{rot} + p$, where d_{trans} and d_{rot} quantify the translation and rotation distances to the goal manipuland pose, and p encompasses conventional penalty components often integrated for RL, such as a penalty for actions and velocities.

r^S is decided by the output of a discriminator, which discriminates whether a state transition belongs to the demonstration distribution or not, and the discriminator is trained together with the policy from scratch. Prior to presenting a state transition as input to the discriminator, an observation map $\Phi(\mathbf{s}_t)$ extracts the features from the system state $\mathbf{s}_t$ for selecting the attributes associated with a particular skill. This is an important aspect since it lets us get away with plans that contain infeasible actions by only mimicking the equilibrium joint poses $\mathbf{q}_a$ for imitation, unlike typical behavior cloning methods that imitate feasible actions.

III. Experiments & Results

We test our approach on a whole-body manipulation task for pivoting and lifting a box, as depicted in Fig. 1(a). This evaluation is conducted using the Punyo robot [42]: an assemblage comprising of two Jaco robot arms, each with soft visuotactile sensors as end-effectors [44] and an array of 7 air-filled pressure-based sensors that cover each arm. To estimate the manipuland pose, we use an OptiTrack motion-capture system.

The code provided in [3] is used for planning. Then, the refined plan obtained from this framework is post-processed for two reasons: (i) to prevent undesired robot-manipuland collisions and (ii) to stabilize the object using a task-specific heuristic that makes the idle left arm to support the box when

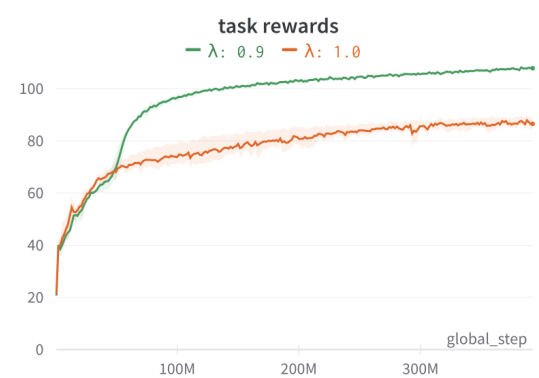


Fig. 3: Performance of PGRL ($\lambda = 0.9$) and RL ($\lambda = 1$) where λ is the task reward weight.

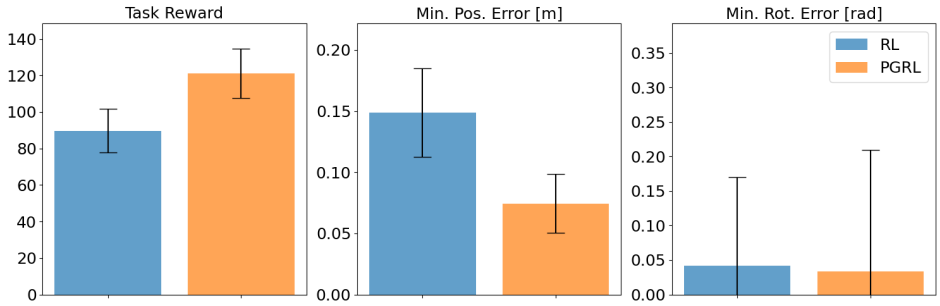


Fig. 4: Aggregated results of rolling out the RL and PGRL policies in 1000 different environments. The box plots show the mean and standard deviation for the task reward and the minimum translational and rotational distances of the manipuland to the goal along each rollout.

moving between segments. For this purpose, the kinematic trajectory optimization method in Drake [45] is used with minimum distance constraints.

For policy learning, we employ the proximal policy optimization (PPO) algorithm [46]. Particularly, we utilize the PPO implementation in [47] and the AMP implementation in [48] to run in Isaac Gym [49]. Furthermore, we incorporate domain randomization [50] to facilitate the transfer of learned behaviors from simulation to real. See the Appendix section for further implementation details.

We run simulation experiments to compare PGRL to GQDP and to standard model-free RL (denoted as RL), which is also trained with PPO using only the task reward. Additionally, we test the policy generated by our proposed approach (PGRL) on hardware. When rolling out the GQDP plan in simulation, the robot encounters difficulty in successfully lifting the box due to the teleportation issue described in section II-A. The inability to achieve success in this scenario in the absence of any model mismatch from planning to simulation prompted the decision not to proceed with testing under conditions of likely model-mismatch in real.

Figure 1 presents a juxtaposition between key points extracted from the plan employed for training and analogous key points identified during the execution of the RL policies. See the accompanying video [Link] for the full motions. The visual analysis shows the proficiency of our method in emulating the motion style imposed by the plan, whereas standard RL leads to unnatural behaviors with unsatisfactory task performance. Specifically, the RL policy can only pivot the box but fails to maintain it at the desired height. Moreover, the RL training in PGRL helps to bridge the gaps in the GQDP plan (i.e., infeasible transitions during the teleports) and enables task completion. It is also noteworthy that PGRL alters the contact sequence in the plan to a more robust one owing to domain randomization.

To compare PGRL and RL, we train policies by maintaining uniformity across all parameters except for λ, which is set to 1 (i.e., no imitation reward) for the RL case. Fig. 3 illustrates the mean task reward curve along with the standard deviation resulting from three training instances with different seeds. Notably, our approach (trained with $\lambda = 0.9$) outperforms RL in terms of task reward even though RL's sole objective is to maximize this metric.

To evaluate the trained policies, we roll them out in 1000 distinct environment instantiations, encompassing variations in the physical properties of the manipuland (such as mass, friction, and size) as well as the initial position (extracted from the identical distribution utilized during training). The mean and standard deviation of the accumulated task reward, and the minimum translational and rotational distances of the manipuland to its goal pose along each rollout for both RL and PGRL are shown in Fig. 4.

IV. Conclusion & Future Work

In this study, we introduced a plan-guided RL framework to convert complex, long-horizon, contact-rich manipulation trajectories to a closed-loop policy that is sufficiently robust to be deployed in the real world. As a proof of concept, we tested the proposed approach on a whole-body box reorientation and lifting task using a single, infeasible plan. The results imply that: (i) PGRL can fix infeasible transitions in the plan while maintaining the desired motion characteristics, (ii) the guidance can help to compose a strategy that is better than one that standard RL can discover alone without extensive reward engineering, and (iii) the RL training and domain randomization can enable obtaining a similar but more robust behavior compared to the plan by altering the contact configurations as necessary.

The preliminary results are promising but there are many open research questions that require further investigation. In particular, our future studies would explore the following:

- incorporating tactile information into the imitation and/or policy observation spaces with the hope that this can mitigate the need for motion capture for tracking the manipuland's pose (which is already restrictive due to heavy occlusions during whole-body interactions) and enable extending the framework to tasks that are more sensitive to force interactions;
- the impact of the quality and quantity of example motions (either plans or teleoperation data with different fidelity) on the performance;
- synthesizing more complex behaviors by using examples for diverse skills with the hope that AMP would be able to automatically compose them;
- understanding the effect of the demonstrated contact sequence on the exploration and how it is modified during the RL training; and
- trying methods like curriculum learning to increase learning efficiency and policy robustness.

REFERENCES

[1] M. T. Mason, "Toward robotic manipulation," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 1–28, 2018.

[2] N. Chavan-Dafle and A. Rodriguez, "Sampling-based planning of in-hand manipulation with external pushes," in *Robotics Research: The 18th International Symposium ISRR*. Springer, 2020, pp. 523–539.

[3] T. Pang, H. Suh, L. Yang, and R. Tedrake, "Global planning for contact-rich manipulation via local smoothing of quasi-dynamic contact models," *arXiv preprint arXiv:2206.10787*, 2022.

[4] X. Cheng, E. Huang, Y. Hou, and M. T. Mason, "Contact mode guided motion planning for quasidynamic dexterous manipulation in 3D," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 2730–2736.

[5] M. Zhang, D. K. Jha, A. U. Raghunathan, and K. Hauser, "Simultaneous trajectory optimization and contact selection for multi-modal manipulation planning," *arXiv preprint arXiv:2306.06465*, 2023.

[6] R. Natarajan, G. L. Johnston, N. Simaan, M. Likhachev, and H. Choset, "Torque-limited manipulation planning through contact by interleaving graph search and trajectory optimization," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 8148–8154.

[7] A. Ö. Önol, R. Corcodel, P. Long, and T. Padır, "Tuning-free contact-implicit trajectory optimization," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 1183–1189.

[8] M. Wang, A. Ö. Önol, P. Long, and T. Padır, "Contact-implicit planning and control for non-prehensile manipulation using state-triggered constraints," in *The International Symposium of Robotics Research*. Springer, 2022, pp. 189–204.

[9] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *arXiv preprint arXiv:2303.04137*, 2023.

[10] S. Haldar, J. Pari, A. Rai, and L. Pinto, "Teach a robot to fish: Versatile imitation from one minute of demonstrations," *arXiv preprint arXiv:2303.01497*, 2023.

[11] M. Du, S. Nair, D. Sadigh, and C. Finn, "Behavior retrieval: Few-shot imitation learning by querying unlabeled datasets," *arXiv preprint arXiv:2304.08742*, 2023.

[12] J. Barreiros, L. Tianshu, M. Chiaramonte, K. Jost, Y. Menguc, N. Colonnese, and P. Agarwal, "Hyfar: A textile soft actuator for haptic clothing interfaces." ACM, 2022.

[13] C. Rognon, S. Mintchev, F. Dell'Agnola, D. Atienza, and D. Floreano, "Flyjacket: An upper body soft exoskeleton for immersive drone control." IEEE, 2018, pp. 2362–2369.

[14] T. Chen, J. Xu, and P. Agrawal, "A system for general in-hand object re-orientation," in *Conference on Robot Learning*. PMLR, 2022, pp. 297–307.

[15] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray *et al.*, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020.

[16] A. Nagabandi, K. Konolige, S. Levine, and V. Kumar, "Deep dynamics models for learning dexterous manipulation," in *Conference on Robot Learning*. PMLR, 2020, pp. 1101–1112.

[17] T. Chen, M. Tippur, S. Wu, V. Kumar, E. Adelson, and P. Agrawal, "Visual dexterity: In-hand dexterous manipulation from depth," in *ICML Workshop on New Frontiers in Learning, Control, and Dynamical Systems*, 2023.

[18] J. EEßerer, N. Bach, C. Jestel, O. Urbann, and S. Kerner, "Guided reinforcement learning: A review and evaluation for efficient and effective real-world robotics," *IEEE Robotics & Automation Magazine*, 2022.

[19] M. Vecerik, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. Riedmiller, "Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards," *arXiv preprint arXiv:1707.08817*, 2017.

[20] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," *arXiv preprint arXiv:1709.10087*, 2017.

[21] X. B. Peng, P. Abbeel, S. Levine, and M. Van de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Transactions On Graphics (TOG)*, vol. 37, no. 4, pp. 1–14, 2018.

[22] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Overcoming exploration in reinforcement learning with demonstrations," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6292–6299.

[23] Y. Zhu, Z. Wang, J. Merel, A. Rusu, T. Erez, S. Cabi, S. Tunyasuvunakool, J. Kramár, R. Hadsell, N. de Freitas *et al.*, "Reinforcement and imitation learning for diverse visuomotor skills," *arXiv preprint arXiv:1802.09564*, 2018.

[24] H. Zhu, A. Gupta, A. Rajeswaran, S. Levine, and V. Kumar, "Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 3651–3657.

[25] V. G. Goecks, G. M. Gremillion, V. J. Lawhern, J. Valasek, and N. R. Waytowich, "Integrating behavior cloning and reinforcement learning for improved performance in dense and sparse reward environments," *arXiv preprint arXiv:1910.04281*, 2019.

[26] S. Christen, S. Stevšić, and O. Hilliges, "Demonstration-guided deep reinforcement learning of control policies for dexterous human-robot interaction," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 2161–2167.

[27] X. B. Peng, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," *arXiv preprint arXiv:2004.00784*, 2020.

[28] A. Nair, A. Gupta, M. Dalal, and S. Levine, "AWAC: Accelerating online reinforcement learning with offline datasets," *arXiv preprint arXiv:2006.09359*, 2020.

[29] S. P. Arunachalam, S. Silwal, B. Evans, and L. Pinto, "Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation," in *2023 ieee international conference on robotics and automation (icra)*. IEEE, 2023, pp. 5954–5961.

[30] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, "Amp: Adversarial motion priors for stylized physics-based character control," *ACM Transactions on Graphics (ToG)*, vol. 40, no. 4, pp. 1–20, 2021.

[31] E. Vollenweider, M. Bjelonic, V. Klemm, N. Rudin, J. Lee, and M. Hutter, "Advanced skills through multiple adversarial motion priors in reinforcement learning," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 5120–5126.

[32] A. Escontrela, X. B. Peng, W. Yu, T. Zhang, A. Iscen, K. Goldberg, and P. Abbeel, "Adversarial motion priors make good substitutes for complex reward functions," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 25–32.

[33] C. Li, M. Vlastelica, S. Blaes, J. Frey, F. Grimminger, and G. Martius, "Learning agile skills via adversarial imitation of rough partial demonstrations," in *Conference on Robot Learning*. PMLR, 2023, pp. 342–352.

[34] C. Li, S. Blaes, P. Kolev, M. Vlastelica, J. Frey, and G. Martius, "Versatile skill control via self-supervised adversarial imitation of unlabeled mixed motions," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 2944–2950.

[35] J. Wu, G. Xin, C. Qi, and Y. Xue, "Learning robust and agile legged locomotion using adversarial motion priors," *IEEE Robotics and Automation Letters*, 2023.

[36] M. Bogdanovic, M. Khadiv, and L. Righetti, "Model-free reinforcement learning for robust locomotion using demonstrations from trajectory optimization," *Frontiers in Robotics and AI*, vol. 9, p. 854212, 2022.

[37] Y. Fuchioka, Z. Xie, and M. Van de Panne, "Opt-mimic: Imitation of optimized trajectories for dynamic quadruped behaviors," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 5092–5098.

[38] A. Miller, S. Fahmi, M. Chignoli, and S. Kim, "Reinforcement learning for legged robots: Motion imitation from model-based optimal control," *arXiv preprint arXiv:2305.10989*, 2023.

[39] D. Kang, J. Cheng, M. Zamora, F. Zargarbashi, and S. Coros, "Rl+ model-based control: Using on-demand optimal control to learn versatile legged locomotion," *arXiv preprint arXiv:2305.17842*, 2023.

[40] L. Wang, Y. Xiang, W. Yang, A. Mousavian, and D. Fox, "Goal-auxiliary actor-critic for 6d robotic grasping with point clouds," in *Conference on Robot Learning*. PMLR, 2022, pp. 70–80.

[41] S. Huang, Z. Wang, P. Li, B. Jia, T. Liu, Y. Zhu, W. Liang, and S.-C. Zhu, "Diffusion-based generation, optimization, and planning in 3d scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 16 750–16 761.

[42] A. Goncalves, N. Kuppuswamy, A. Beaulieu, A. Uttamchandani, K. M. Tsui, and A. Alspach, "Punyo-1: Soft tactile-sensing upper-body robot for large object manipulation and physical human interaction," in *2022 IEEE 5th International Conference on Soft Robotics (RoboSoft)*. IEEE, 2022, pp. 844–851.

[43] S. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *Research Report 9811*, 1998.

[44] N. Kuppuswamy, A. Alspach, A. Uttamchandani, S. Creasey, T. Ikeda, and R. Tedrake, "Soft-bubble grippers for robust and perceptive manipulation," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 9917–9924.

[45] R. Tedrake and the Drake Development Team, "Drake: Model-based design and verification for robotics," 2019. [Online]. Available: https://drake.mit.edu

[46] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[47] D. Makoviichuk and V. Makoviychuk. rl-games: A high-performance framework for reinforcement learning. [Online]. Available: https://github.com/Denys88/rl_games

[48] IsaacGymEnvs. [Online]. Available: https://github.com/NVIDIA-Omniverse/IsaacGymEnvs

[49] 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.

[50] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2017, pp. 23–30.

V. Appendix

A. Implementation Details

1) Task Reward: The task reward is implemented as

$$r_t^G = w_{trans}(1/(\|d_{trans}(\mathbf{q}_t^u, \mathbf{q}_{goal}^u)\| + 0.1)) + w_{rot}(1/(\|d_{rot}(\mathbf{q}_t^u, \mathbf{q}_{goal}^u)\| + 0.1)) + w_{action}||\mathbf{a}_t||^2 + w_{velocity}||\dot{\mathbf{q}}_t^u||^2 + w_{termination}\mathbb{1}(\mathbf{q}_t^u).$$

The initial two terms incentivize task completion. The following two terms impose penalties on both the robot's actions and the manipuland's velocity. The final term enforces a penalty upon the activation of termination conditions. These conditions manifest when the box deviates significantly from the center of the table or experiences a drop. The function $\mathbb{1}(\cdot)$ is an activation function based on termination conditions. We use the weights $w_{trans} = 0.07$, $w_{rot} = 0.03$, $w_{action} = -0.002$, $w_{velocity} = -0.002$, and $w_{termination} = -1$ for all experiments. The task is defined by $\mathbf{q}_{goal}^u = (0.15, 0, 0.4, 0, -\pi/2, 0)$ concatenating the positions in m and roll-pitch-yaw in rad.

2) Observations: The observation of the discriminator is joint position transitions of the robot ($\mathbf{q}_t^a$, $\mathbf{q}_{t+1}^a$). The observation of the policy is ($\mathbf{q}_t^a, \mathbf{q}_t^u, \mathbf{p}_t^{ee}$) where $\mathbf{p}_t^{ee}$ is the Cartesian end-effector poses and $\mathbf{q}_t^u$ is the manipuland pose.

3) Training Parameters: The learning networks and algorithm were implemented in PyTorch 1.8.1 with CUDA 12.0. The training procedure encompassed the collection of experiences from 4096 uncorrelated instances of the simulator performed in parallel. The entirety of the experimental work was executed on a desktop equipped with NVIDIA 3090 GPUs. A single run comprising 1,500 iterations, adhering to the aforementioned computational settings and device specifications, was accomplished within approximately 4 hours. Detailed training parameters and network architectures are outlined in Tables I and Table II, respectively.

Parameter	Value
γ	0.99
τ	0.95
λ	0.9
parallel training environments	4096
sample per update iteration	4096
batch size	512
learning rate	$5e^{-5}$
KL divergence target	$8e^{-3}$
clip range	0.2
horizon length	64
discriminator weight decay	$1e^{-4}$
discriminator gradient penalty	10

TABLE I: Training parameters.

Network	Type	Hidden Layers	Activation
policy	MLP	[256, 128, 64]	Rectified Linear Unit (ReLU)
value function	MLP	[256, 128, 64]	Rectified Linear Unit (ReLU)
discriminator	MLP	[256, 128, 64]	Rectified Linear Unit (ReLU)

TABLE II: Network architecture.

4) Domain Randomization: Domain randomization techniques are judiciously employed during the training process to enhance robustness and performance. This involves the systematic application of disturbances to various parameters. Specifically, we initialize the manipuland on the table uniformly within a 5 cm radius of its nominal initial position [0.35. 0, 0.13] m. Additionally, a perturbation sampled from the uniform distribution $\mathcal{U}(-0.05, 0.05)$ is added to its yaw angle. A noise sampled from $\mathcal{N}(0, 0.02)$ rad is injected into the robot's actions (i.e., joint position commands to a stiffness controller), and a disturbance drawn from $\mathcal{U}(0.0, 0.5)$ m/s^2 is added to the gravity. Furthermore, the distributions $\mathcal{U}(0.9, 1.1)$, $\mathcal{U}(0.8, 1.2)$, and $\mathcal{U}(0.8, 1.0)$ are used to scale the dimensions and mass of the manipuland and the friction coefficient for all contacts. The nominal manipuland dimensions and mass are [0.41. 0.315, 0.26] m and 0.45 kg, and the nominal friction coefficient is 1.

B. Acknowledgments

The authors extend their sincere appreciation to Tao Chen, Chenhao Li, and Binghao Huang for their valuable assistance in the implementation process. Gratitude is also expressed to Russ Tedrake, Hongkai Dai, Benjamin Burchfiel, and the Dexterous Manipulation group at TRI for their insightful discussions. Furthermore, the authors would like to thank Alex Alspach, Sam Creasey, Aimee Goncalves, and the entire Punyo team for their generous support, constructive feedback, and the exceptional hardware provided.