Making use of design-aware policy optimization in legged-robotics co-design

Gabriele Fadini, Stelian Coros
ETH Zürich, Computational Robotic Lab (CRL), Switzerland
Corresponding author: gfadini@ethz.ch

Abstract: Our ongoing research aims to investigate the potential of integrating robot design optimization with reinforcement learning (RL). In co-design literature, exploiting the ties between design and control seems to be the key to unlock otherwise unreachable performance. However, the problem of obtaining policies that well adapt to a range of different robots are still open. In this extended abstract, we would like to reason about the challenges that the policy optimization problem in this setting brings. Moreover, we hint at a few possible future research directions that may help in advancing of robot morphology and design-aware control policies.

Keywords: CoRL, Robots, Learning

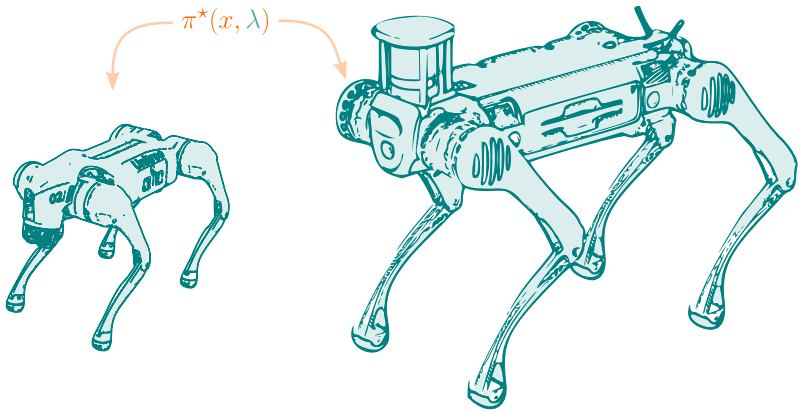

Figure 1: Even if the behaviors of robots with similar morphology is quite consistent (here Go2 and B2 from Unitree), the optimal policy $\pi^\star(x)$ needs also to consider design-specific parameters λ. Co-design needs to reason about hardware specificities too. Can we recover policies that generalize for different hardware and use them in this optimization?

1 Why co-optimizing behaviour and design?

Traditionally, the field of robotics has treated design and control as separate processes. This separation has been shown to generally lead to suboptimal and brittle solutions, often requiring several stages of refinement to achieve an adequate performance. This is especially true in the realm of robot design. Once the physical implementation of a platform is chosen, it may happen that some potentialities in terms of motions, reach, task requirements and performance may not be attainable anymore. Conversely, designing concurrently both control and plant, though looks like a more complex but beneficial way to determine the embodiment of a system: we may tailor a robot for the specific requirements from the on-set. This approach is broadly called co-design. The simultaneous optimization of a robot's morphology and control policy represents a paradigm shift in robotics engineering. In traditional robotics, we typically focus on optimizing robot behavior for new tasks,

while relying on human-designed, multipurpose robot bodies. However, nature demonstrates that specialized body morphologies can lead to superior performance in specific tasks. This suggests that co-evolving both robot morphology and behavior could result in more efficient and task-specific robotic systems. Co-design, if compared to traditional design, shows the potential benefits of unifying control and design. This is true for a multitude of different systems and tasks. In recent literature it has been proven several times [1, 2, 3, 4, 5, 6, 7] to lead to optimized systems that can act better for a given goal task. From simple underactuated systems [8], to UAV [9] and humanoids [10]. This approach promises to yield more efficient and adaptable robotic systems by considering the intricate interplay between physical structure and behavior. However, implementing design-aware policy optimization in this context presents several unique challenges and requires innovative techniques.

2 Optimization-based co-design

Several optimization frameworks combining trajectory and hardware optimization are proposed to implement design-aware co-design. These frameworks utilize genetic algorithms for hardware optimization and trajectory optimization techniques for optimal control selection [7]. The process begins with parameterizing the robot's design, including its geometry, materials, and actuator properties. These parameters could be either optimized in a monolithic optimization framework, leveraging sensitivities to the robot dynamics and the model, or through some sampling-based techniques (such as genetic algorithms) in bi-level optimization schemes.

Within this framework, a powerful tool to allow optimization of tasks is given by differentiable physics engines. This is crucial for gradient-based optimization as it allows for efficient backpropagation through the entire system, from task performance metrics to design variables. The physics engine should support various material properties and environmental interactions to accurately model soft robots and their surroundings. One of the main limitations of the method is that usually it becomes tied to a specific design task. While it is possible to generalize it to more tasks, [11], this comes at an increased computational cost and requires also use-specific insights to refine the solution (in the form of a Pareto front). Moreover, in practical applications, there are several sources of perturbations which cannot be taken readily into account in the trajectory optimization phase (e.g. noise, friction, delays). Coming up with methods to mitigate the impact of the design choices and plan is crucial. Some research has been done in the past, showing good performance concerning noise rejection in stochastic optimization [5] and with a-posteriori estimation of the performance of the design in simulation [6].

3 Learning-based co-design

In some cases, such as in the domain of soft robotics, the sheer number of degrees of freedom makes the use of trajectory optimization extremely complex. In this domain, an incredibly powerful tool to obtain control policy is given by reinforcement learning [12]. A key component of the implementation is the development of a latent state representation that captures salient features of the robot's dynamics [12]. This representation can be learned using deep variational convolutional autoencoders, which compress the high-dimensional state space into a lower-dimensional latent space. This learned representation facilitates more efficient policy optimization by focusing on task-relevant features. Some extensions have also been considered to tackle co-design problem allowing an adaptation of the design together to the task satisfaction [13, 14, 15, 16]. Leveraging the sampling of actions in a parallelized way it is possible to achieve policies that generalize well and can also be deployed on the real system zero-shot. This led to incredible achievements on high-degrees of freedom systems, with complex tasks, possibly including contact and environment interactions. Another remarkable advantage of the method is that it works also purely based on sensory data from the robot, allowing the generation of complex pipelines directly from data end-to-end.

Exploiting Differentiable Information Within this realm, a further advantage is given by policy optimization techniques which exploit also simulation derivatives to speed up training [17, 18, 19,

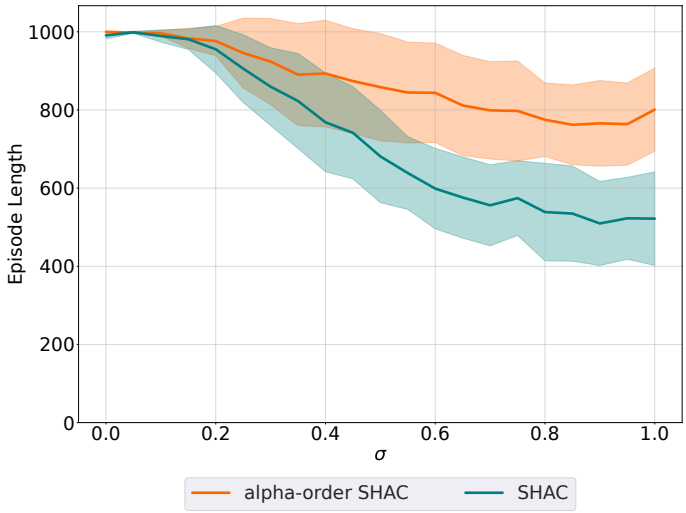

Figure 2: In a contact-rich locomotion environment (Ant), the use of α-order gradient estimation leads to more robust policies that behave better with respect to action noise $\mathcal{N}(0, \sigma)$. This is shown in this figure by the significantly higher episode lengths if compared to vanilla SHAC.

20, 21, 3]. The speed-up of these policy optimization techniques is remarkable, however, as they rely on specific approximations of the contact-rich environments. At the current state, they may require some further tuning to be suited to complex robotic applications such as loco-manipulation problems [22, 23] reduce the sim-to-real gap. In our ongoing research, we have experimentally seen a sharp degradation of the optimal policy performance, when facing slightly different scenarios. As a possible solution, the use of sharpness-aware optimization and α-order estimation of the gradients have shown improvements in the generalization of the policies. Focusing on the latter case, the idea is to introduce de-biasing information about the contact problem also from zero order estimation. Some preliminary results are shown in Fig. 2 where the policy obtained with Short-Horizon Actor-Critic (SHAC) does not generalize to different levels of noise σ. Higher episode lengths mean that the performance remains stable for longer and does not meet early terminations. Our empirical tests have shown that some improvement can be obtained with the introduction of zero-order information. However further studies are ongoing to understand the proper trade-off of the adaptation of α in specific environments and tasks.

Despite these improvements, one of the downsides of this approach is not to scale up with the number of design parameters. Training a policy is a much more complex problem. In the frame of co-design, learning policies that adapt different designs of the robot require a careful augmentation of the state or action space to include these additional parameters in order to condition the policy with respect to changes in the design. This comes with the downsides of increased computational costs, the need for much more samples and the classical pitfalls of training complex models.

4 Similarities and shortcomings of the two approaches

The two families of methods: optimization-based and learning-based have plenty of points in common, as they try to solve the same problem. In this section the strengths and shortcomings of each are briefly discussed. We suppose that we have a unique representation of the problem at hand (unifying notation and problem description to match the trajectory optimization one i.e. "cost minimization" vs "reward maximization"). Once this step is done, the similarities are exemplified in Fig. 3.

Trajectory optimization allows us to obtain a specific, locally optimal trajectory, building on well-established non-linear optimization techniques. This comes with the advantage of having tailored and fast optimization techniques which can potentially also include constraints and, to a given extent,

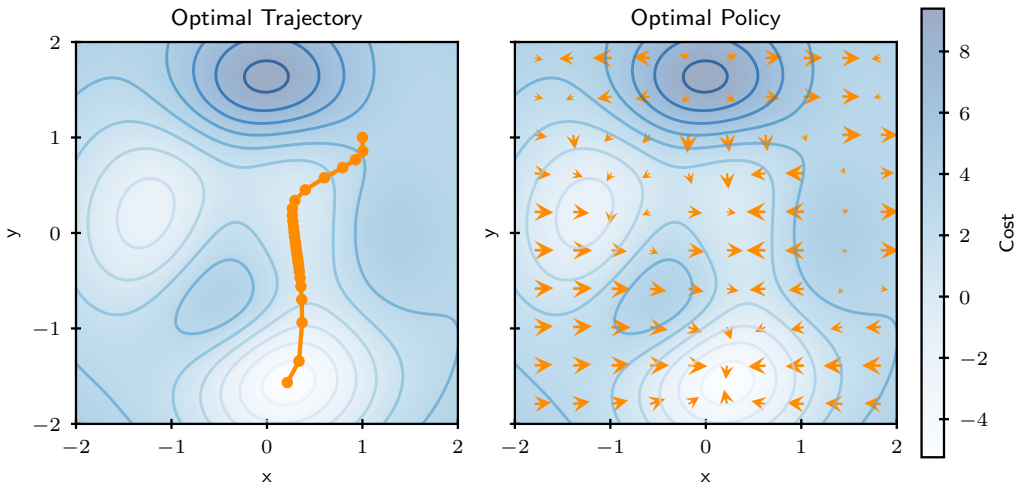

Figure 3: Similarities between trajectory optimization and policy optimization. Trajectory optimization gives a specific trajectory, from a specific state that brings to the optimum. Learning provides a more general rule that generalize in the whole state space and can recover optimal trajectories.

guarantees. However, while trajectory optimization provides solutions that minimize a cost function with constraints satisfaction, they do not generalize to the whole state space this is visualized again in Fig.3 the optimal plan is the one that goes to the global minimum from a given starting condition. It is possible to understand how the optimal trajectory will be impacted by the modification of the problem parameters with sensitivity analysis. The optimization-based approach is outputting only the best set of local controls and designs to fulfill the task, any modification and refinement requires running new computations. The optimal control is also tied to planning for an ideal environment and, as previously stated, it may be difficult to introduce generalizations to it. On top of this, finding a general controller to track the optimal trajectory is solved at a later stage and usually either requires tuning or on-line approaches, such as MPC. Trying to solve the co-design problem together with the controller-planner may be a way to hard to tackle. Due to the problem's dimensionality, while it is relatively easy to adapt optimal policy to new designs, exploiting sensitivities, and obtaining optimal policies for each design may reveal to be a much harder problem.

Policy learning 's goal is to find a policy that works in the robot's state space and hence is inherently more general, as it automatically encompasses for the planner and controller in the real-world environment. Theoretically, with an adequate amount of learning, any robotic task can be achieved. In the same problem setting, the optimal policy can automatically recover optimal trajectory plans, and at the same time be used on the system after an off-line training from experience. However, in robotics the state space is continuous and this is just leading to approximations of the optimal policy. This may be problematic in the setting of co-design if finding the optimal design or the optimal policy is of interest. Differently to optimization-based methods, there is more difficulty in assessing optimality, but higher generalization potential. Optimal actions are recovered in the whole state space, as shown in Fig. 3. One may argue that domain randomization (for friction, inertias, etc.) alone shows that it is possible to select a policy which does not overfit for specific parameters but rather can be applied to several systems. A fundamental technique in design-aware policy optimization is the formulation of a unified state space that encompasses both morphological and control parameters. This expanded representation allows for the exploration of how changes in physical design influence control strategies and vice versa. Implementing this unified state space requires careful consideration of how to effectively encode diverse physical properties (such as link lengths, joint types, or material properties) alongside traditional control state variables. In this setting, several meta-learning techniques have been recently proposed. Other works focus on the generation

of meta-policies capable of several papers that explore the concept of meta-policies in robotics and reinforcement learning. Meta-policies are higher-level policies that can adapt to different robot morphologies or task conditions. The key points about these works is to develop policies that can work across different robot designs or body configurations without needing to be retrained from scratch. In [24] model-free meta-reinforcement learning is used to train a locomotion policy that can quickly adapt to different designs and achieves close-to-optimal performance for each design instance after adaptation. AnyMorph [25] presents a novel approach to reinforcement learning by enabling the transfer of policies across different agent morphologies without the need for prior morphological descriptions. This is achieved through a data-driven method that learns morphology representations directly from the reinforcement learning objectives, enhancing zero-shot generalization to new agents. The methodology contrasts with traditional approaches that require hand-designed morphology descriptions, thus streamlining the learning process. MorAL [26] concurrently trains a control policy alongside an adaptive module that takes into account the robot's temporal states. This module allows the control policy to implicitly identify the properties of different robot platforms and estimate body velocity in real-time. Extensive experiments conducted in both real-world and simulated environments demonstrate that this controller enables robots with significantly different morphologies to effectively navigate a variety of harsh indoor and outdoor terrains. Other work, to generate locomotion policies is also found in recent literature extending similar concepts to robot of similar morphology [27, 28, 29]. In general, there are several ways to make the loss function agnostic for the model and ultimately it also boils down to the specific requirements, however, it may be difficult to keep the policy generation unbiased.

Main take Neither of the two classes of optimizations is completely fit to generalize *close-to-optimal* and *unbiased* policies for a class of systems. On one side, trajectory optimization provides a poorer generalization as it solves a specific optimal control problem instance, while on the other side, learning may generalize better but the optimality of its solutions and the bias introduced in meta-learning may be detrimental. However, by exploiting the strengths of each of these methods, we can obtain policies that are more adaptable to the co-design problem. In particular, by looking at some trajectories generated by bi-level planners, one can notice plenty of structure and similarities that should be simple to generalize. Moreover, the optimality of the policy can be reintroduced in the policy directly by imitation of optimal trajectories. In the next section, we will discuss methods to bridge the gap between the two in this setting.

5 Design-aware policy optimization via hybrid approaches

Imitation Some techniques that have still to be fully explored involve the use of hybrid approaches. For instance, one possibility is the use of supervised learning of optimal trajectories via the synthesis of control policies via imitation learning [30]. In imitation learning, one possible way to generalize learning with different robot designs is via spatiotemporal retargeting [31], this method shows promise in generalizing the learning of behaviors on legged robots. Taking inspiration from this result, it may be possible to extend the same reasoning also considering optimal trajectories as expert demostrations.

In co-design One of the primary challenges in this domain is the vastly increased search space resulting from the combination of design and control parameters. To address this, hierarchical optimization structures have shown promise. These approaches typically employ a higher-level algorithm, such as evolutionary strategies or Bayesian optimization, to explore a (small) morphological design space, while lower-level algorithms optimize control policies for each candidate design [32]. SoftZoo is a co-design framework for soft robots [33] that exploits differential information. In such framework several robot design can be evaluated and optimized in a variety of environments. This work leverages a reinforcement learning policy to then improve the design via a Bayesian approach. Other work, related to hybrid methods, bridges the gap between trajectory optimization and RL by using TO to guide the exploration of an actor-critic RL algorithm, improving sample efficiency [34, 35]. Morevoer, in this work also first-order information about the optimal trajectories could be exploited via Sobolev learning a technique that uses the information of sensitivities to improve the approximation of the

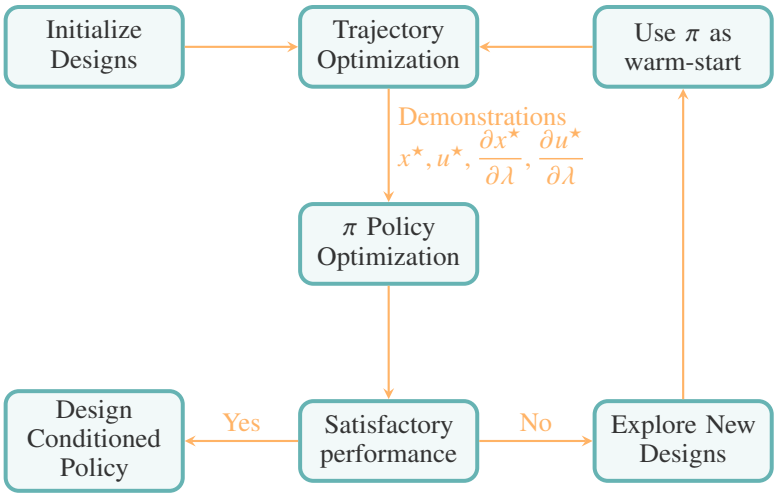

Figure 4: Sampling-based policy optimization with trajectory optimization

learning network for the policy [36]. Even if not explicitly targeted to a co-design application, this work shows how differential information can be used from trajectory optimization.

6 Design-aware policy optimization

Considering all of these results, we plan to extend co-design pipelines in the future to better benefit of reinforcement learning and differentiable simulation. We sum up a sketch of such an implementation which we plan to test in the future. In our ongoing work, we are currently using differentiable simulation information via [37, 38], recovering, after trajectory optimization methods, optimal state and control trajectories $x^\star, u^\star$. Thanks to the differentiability of the framework, also the sensitivities of these trajectories with respect to the design parameters λ can be recovered. Using standard optimization techniques, constraints on the trajectories (even non-Markovian ones [11]) can be enforced. These trajectories can be used to inform design-dependent policies π in a setting similar to Sobolev learning. We envision a method similar to what is shown in Fig. 4. At the beginning, initial samples from the design space are selected, and then for each, a trajectory optimization problem is solved from a set of initial conditions (which we assume fixed e.g. in the joint space). The optimal solutions and their sensitivities are then used as experts to inform a design-aware policy $\pi(x, \lambda)$ which will have to match the optimal trajectories and rewards in the same problem setting. For this step to be achievable, the same simulation of the dynamics and reward formulation is enforced by the use of the same differentiable simulator. If the optimized policy provides a satisfactory performance (for instance e.g. against validation data) then the optimization can terminate. In the case the optimization is unsuccessful (the policy is not generating close to optimal trajectories in the computational budget), then we deem the sampled design insufficient for proper generalization. As a next step, to achieve higher generalization new data is generated trying to provide more information about the design space. This step can be done by trying to maximize the difference concerning previously selected designs, similarly to what is common in entropy-based methods. For each of the new design samples, an optimization problem is solved obtaining again optimal trajectories stored in the dataset of demonstrations. All the demonstrations are optimal, so the data is simply stored and reused in the next run of the policy optimizer. This loop continues until the policy generalizes in the design space. As a last step this policy can be used for co-design purposes as in [32] or even in gradient-based approaches exploring the differentiability with respect to the design parameters in simulation.

7 Conclusions and future work

In conclusion, while design-aware policy optimization in robot co-design offers exciting possibilities for creating more capable and efficient robots, it also presents a host of technical challenges. Balancing exploration and exploitation between these levels is crucial and remains an active area of research. Another key technique is the development of design-aware loss functions that capture the complex interplay between morphology and control. Balancing the trade-off between evaluation accuracy and computational efficiency remains an ongoing challenge. We have indicated a possible way to complement nicely the computational speed-up from gradient-based methods and differentiable simulation and learning. We plan to explore more of these hybrid methods in future research paving the way for more adaptable policies that can better express the robot's properties for guiding the design phase. To tackle the open challenges of co-design in robotics, interdisciplinary approaches are needed from the start, drawing from fields such as reinforcement learning, evolutionary computation, multi-objective optimization, down-the-line prototyping, and advanced manufacturing. As research in this area progresses, we can expect to see the development of more sophisticated techniques that push the boundaries of what's possible in robotic design and control. One of the main open problems is the optimization problem dimensionality. These techniques may perform well for a moderate number of variables, but may become a bottleneck for complex designs requiring hundreds of parameters. The development of design-aware loss functions that capture the interplay between morphology and control is necessary. And we believe it could be boosted by integrating first-order information into learning.

References

[1] S. Ha, S. Coros, A. Alspach, J. Kim, and K. Yamane. Joint optimization of robot design and motion parameters using the implicit function theorem. In *Robotics: Science and Systems XIII*. Robotics: Science and Systems Foundation. ISBN 978-0-9923747-3-0. doi:10.15607/RSS.2017.XIII.003. URL http://www.roboticsproceedings.org/rss13/p03.pdf.

[2] M. Geilinger, R. Poranne, R. Desai, B. Thomaszewski, and S. Coros. Skaterbots: optimization-based design and motion synthesis for robotic creatures with legs and wheels. 37(4):1–12. ISSN 0730-0301, 1557-7368. doi:10.1145/3197517.3201368. URL https://dl.acm.org/doi/10.1145/3197517.3201368.

[3] M. Geilinger, D. Hahn, J. Zehnder, Bächer, et al. ADD: Analytically differentiable dynamics for multi-body systems with frictional contact, 2020. URL http://arxiv.org/abs/2007.00987.

[4] G. Grandesso, G. Bravo Palacios, P. Wensing, M. r. Fontana, and A. Del Prete. Exploring the limits of a hybrid actuation system through co-design. doi:10.13140/RG.2.2.28566.78400.

[5] G. Bravo Palacios, A. Del Prete, and P. Wensing. One robot for many tasks: Versatile co-design through stochastic programming. PP:1–1. doi:10.1109/LRA.2020.2969948.

[6] G. Fadini, T. Flayols, A. D. Prete, and P. Soueres. Simulation aided co-design for robust robot optimization. 7(4):11306–11313, . ISSN 2377-3766, 2377-3774. doi:10.1109/LRA.2022.3200142. URL https://ieeexplore.ieee.org/document/9863656/.

[7] G. Fadini, T. Flayols, A. Del Prete, N. Mansard, and P. Soueres. Computational design of energy-efficient legged robots: Optimizing for size and actuators. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9898–9904. IEEE, . ISBN 978-1-72819-077-8. doi:10.1109/ICRA48506.2021.9560988. URL https://ieeexplore.ieee.org/document/9560988/.

[8] F. Girlanda, L. Shala, S. Kumar, and F. Kirchner. Robust co-design of canonical underactuated systems for increased certifiable stability. In *2024 IEEE International Conference on Robotics*

and Automation (ICRA), pages 13271–13277. doi:10.1109/ICRA57147.2024.10611645. URL http://arxiv.org/abs/2403.10966.

[9] F. Bergonti, G. Nava, V. Wüest, A. Paolino, G. L'Erario, D. Pucci, and D. Floreano. Co-design optimisation of morphing topology and control of winged drones. doi:10.48550/ARXIV.2309.13948. URL https://arxiv.org/abs/2309.13948. Publisher: arXiv Version Number: 1.

[10] C. Sartore, L. Rapetti, and D. Pucci. Optimization of humanoid robot designs for human-robot ergonomic payload lifting. In *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*, pages 722–729. IEEE. ISBN 9798350309799. doi:10.1109/Humanoids53995.2022.10000222. URL https://ieeexplore.ieee.org/document/10000222/.

[11] G. Fadini, S. Kumar, R. Kumar, T. Flayols, A. Del Prete, J. Carpentier, and P. Souères. Co-designing versatile quadruped robots for dynamic and energy-efficient motions. 42(6):2004–2025. ISSN 0263-5747, 1469-8668. doi:10.1017/S0263574724000730. URL https://www.cambridge.org/core/product/identifier/S0263574724000730/type/journal_article.

[12] A. Spielberg, A. Zhao, Y. Hu, T. Du, W. Matusik, and D. Rus. Learning-in-the-loop optimization: End-to-end control and co-design of soft robots through learned deep latent representations. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d. Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/438124b4c06f3a5caffab2c07863b617-Paper.pdf.

[13] T. Chen, Z. He, and M. T. Ciocarlie. Hardware as policy: Mechanical and computational co-optimization using deep reinforcement learning. In *Conference on Robot Learning*. URL https://api.semanticscholar.org/CorpusID:221095482.

[14] C. Schaff. Neural approaches to co-optimization in robotics. doi:10.48550/ARXIV.2209.00579. URL https://arxiv.org/abs/2209.00579. Publisher: arXiv Version Number: 1.

[15] A. Gupta, S. Savarese, S. Ganguli, and L. Fei-Fei. Embodied intelligence via learning and evolution. 12(1):5721. ISSN 2041-1723. doi:10.1038/s41467-021-25874-z. URL https://www.nature.com/articles/s41467-021-25874-z.

[16] K. S. Luck, R. Calandra, and M. Mistry. What robot do i need? fast co-adaptation of morphology and control using graph neural networks. URL http://arxiv.org/abs/2111.02371.

[17] J. Xu, V. Makoviychuk, Y. Narang, F. Ramos, W. Matusik, A. Garg, and M. Macklin. Accelerated policy learning with parallel differentiable simulation. URL http://arxiv.org/abs/2204.07137.

[18] I. Georgiev, K. Srinivasan, J. Xu, E. Heiden, and A. Garg. Adaptive horizon actor-critic for policy learning in contact-rich differentiable simulation. URL http://arxiv.org/abs/2405.17784.

[19] M. A. Z. Mora, M. Peychev, S. Ha, M. Vechev, and S. Coros. PODS: Policy Optimization via Differentiable Simulation. In *Proceedings of the 38th International Conference on Machine Learning*, pages 7805–7817. PMLR, July 2021. URL https://proceedings.mlr.press/v139/mora21a.html. ISSN: 2640-3498.

[20] Y. Song, S. Kim, and D. Scaramuzza. Learning quadruped locomotion using differentiable simulation. URL http://arxiv.org/abs/2403.14864.

[21] J. Y. Luo, Y. Song, V. Klemm, F. Shi, D. Scaramuzza, and M. Hutter. Residual policy learning for perceptive quadruped control using differentiable simulation. URL http://arxiv.org/abs/2410.03076.

[22] R. Antonova, J. Yang, K. Murthy, et al. Rethinking optimization with differentiable simulation from a global perspective, 2022. URL https://arxiv.org/abs/2207.00167.

[23] H. J. T. Suh, M. Simchowitz, K. Zhang, et al. Do differentiable simulators give better policy gradients?, 2022. URL https://arxiv.org/abs/2202.00817.

[24] A. Belmonte-Baeza, J. Lee, G. Valsecchi, and M. Hutter. Meta reinforcement learning for optimal design of legged robots. *IEEE Robotics and Automation Letters*, 7(4):12134–12141, 2022. ISSN 2377-3766, 2377-3774. doi:10.1109/LRA.2022.3211785. URL http://arxiv.org/abs/2210.02750.

[25] B. Trabucco, M. Phielipp, and G. Berseth. AnyMorph: Learning transferable polices by inferring agent morphology, 2022. URL http://arxiv.org/abs/2206.12279.

[26] Z. Luo, Y. Dong, X. Li, R. Huang, Z. Shu, E. Xiao, and P. Lu. MorAL: Learning morphologically adaptive locomotion controller for quadrupedal robots on challenging terrains. *IEEE Robotics and Automation Letters*, 9(5):4019–4026, 2024. ISSN 2377-3766, 2377-3774. doi:10.1109/LRA.2024.3375086. URL https://ieeexplore.ieee.org/document/10463132/.

[27] G. Feng, H. Zhang, Z. Li, X. B. Peng, B. Basireddy, L. Yue, Z. Song, L. Yang, Y. Liu, K. Sreenath, and S. Levine. GenLoco: Generalized locomotion controllers for quadrupedal robots, 2022. URL http://arxiv.org/abs/2209.05309.

[28] A. Gupta, L. Fan, S. Ganguli, and L. Fei-Fei. MetaMorph: Learning universal controllers with transformers, 2022. URL http://arxiv.org/abs/2203.11931.

[29] F. D. Giuro, F. Zargarbashi, J. Cheng, D. Kang, B. Sukhija, and S. Coros. Meta-reinforcement learning for universal quadrupedal locomotion control, 2024. URL http://arxiv.org/abs/2407.17502.

[30] 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. *IEEE Robotics and Automation Letters*, 8(10):6619–6626, 2023. doi:10.1109/LRA.2023.3307008.

[31] T. Yoon, D. Kang, S. Kim, M. Ahn, J. Cheng, S. Coros, and S. Choi. Spatio-temporal motion retargeting for quadruped robots. URL http://arxiv.org/abs/2404.11557.

[32] F. Bjelonic, J. Lee, P. Arm, D. Sako, D. Tateo, J. Peters, and M. Hutter. Learning-based design and control for quadrupedal robots with parallel-elastic actuators. 8(3):1611–1618. ISSN 2377-3766, 2377-3774. doi:10.1109/LRA.2023.3234809. URL https://ieeexplore.ieee.org/document/10008034/.

[33] T.-H. Wang, P. Ma, A. E. Spielberg, Z. Xian, H. Zhang, J. B. Tenenbaum, D. Rus, and C. Gan. SoftZoo: A soft robot co-design benchmark for locomotion in diverse environments. URL http://arxiv.org/abs/2303.09555.

[34] E. Alboni, G. Grandesso, G. P. Rosati Papini, J. Carpentier, and A. Del Prete. Cacto-sl: Using sobolev learning to improve continuous actor-critic with trajectory optimization. In *Learning for Dynamics and Control Conference (under review)*, 2024.

[35] G. Grandesso, E. Alboni, G. P. Papini, P. M. Wensing, and A. Del Prete. Cacto: Continuous actor-critic with trajectory optimization - towards global optimality. *IEEE Robotics and Automation Letters*, 8(6):3318–3325, 2023. ISSN 23773766. doi:10.1109/LRA.2023.3266985.

[36] W. M. Czarnecki, S. Osindero, M. Jaderberg, G. Świrszcz, and R. Pascanu. Sobolev training for neural networks. URL http://arxiv.org/abs/1706.04859.

[37] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012. doi:10.1109/IROS.2012.6386109.

[38] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/jax-ml/jax.