A Modular Software Ecosystem for Research in Learning-Based Receding Horizon Control

Andrea Patrizi*†, Carlo Rizzardo*, Arturo Laurenzi * and Nikos G. Tsagarakis*

***Abstract*— Robotics research in manipulation and locomotion is undergoing a transformative shift towards the use AI-based tools, where purely learning-based control policies and pipelines are starting to take over the field. Despite being shown to be capable of remarkable robustness and performance even when applied to real-world environments, some key inherent limitations such as safety guarantees, interpretability, etc.. persist. For this reason, it is the authors' belief that more classical control approaches should not be considered outdated yet. We thus advocate for a hybrid approach, combining offline data-based policy design, specifically through Reinforcement Learning (RL), with classical online Motion Planning, i.e. Receding Horizon Control (RHC). Even though this kind of hybrid approaches are not entirely new, to the authors' knowledge, there is no specific tool currently available for search in this domain. To this purpose, we developed a modular software ecosystem, hereby synthetically presented in its main components and features. We care to stress that the framework is currently under active development, and features might not be stable or could be lacking. To facilitate usability and diffusion, we made all the core components open source under the GPLv2 license. To showcase the potential of the framework, we furthermore briefly present a proof-of-concept example combining a high-level RL agent coupled with a lower-level MPC controller for the execution of a simple locomotion task on a hybrid wheeled-legged quadruped robot.**

I. A Hystorical Overview: *from adaptive control and dynamic programming to modern Receding Horizon Control and Reinforcement Learning*

[1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15]

[16]

[17] [18] [19] [20]

II. RL versus RHC: *formulation*

III. RL versus RHC: *shortcomings and advantages*

IV. A hybrid approach: *Learning-Based Receding Horizon Control with RL*

V. Implementation: *available tools, framework overview and rationale*

[21] [22] [23] [24] [25]

VI. A proof-of-concept locomotion example: *learning acyclic stepping*

† Department of Informatics, Bioengineering, Robotics and Systems Engineering, Università di Genova, Via All'Opera Pia 13, 16145 Genova.

* Humanoids and Human-Centred Mechatronics (HHCM), Istituto Italiano di Tecnologia (IIT), Via San Quirico 19d, 16163 Genova.

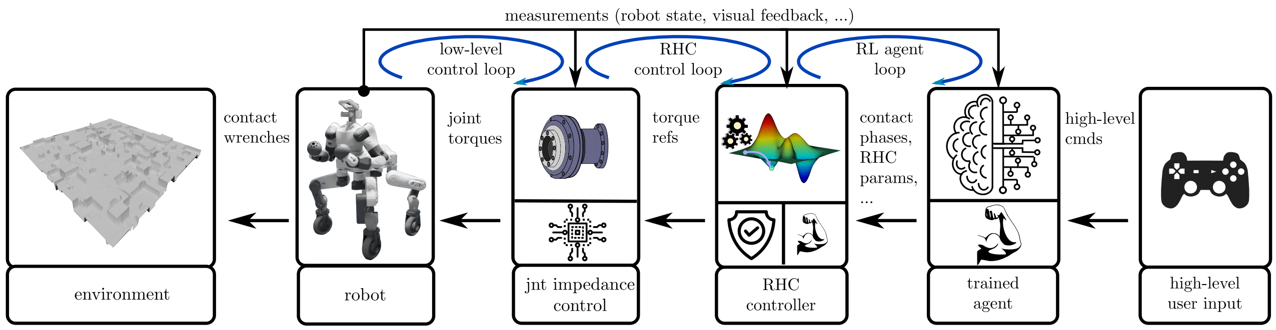

Fig. 1. Our take on Learning-based Receding Horizon Control: a MPC controller is exposed to a RL agent through key runtime parameters, like contact phases, its internal state (costs, constrains..) and interfaces for setting task commands. The agent learns to exploit the underlying RHC controller to perform the tracking of user-specified high-level task references. This allows to both tackle problems which are non-trivial at the MPC level (like phase selection), while also exploiting the flexibility of the agent to complete tasks and the capability of the MPC of ensuring safety.

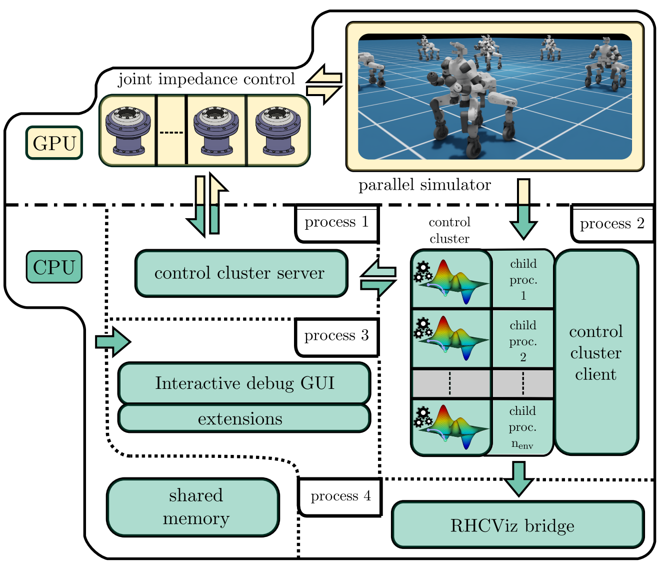

Fig. 2. High-level overview of the training environment to which the agent is exposed: the robot in the simulator is controlled through a joint-level impedance controller, which is in turn used by a higher-level receding horizon controller. The agent can indirectly control the robot through the latter.

References

[1] R. Bellman, "A markovian decision process," *Journal of mathematics and mechanics*, pp. 679–684, 1957.

[2] R. Bellman and R. Kalaba, "Dynamic programming and adaptive processes: mathematical foundation," *IRE Transactions on Automatic Control*, no. 1, pp. 5–10, 1960.

[3] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[4] S. M. Kakade, "A natural policy gradient," *Advances in neural information processing systems*, vol. 14, 2001.

[5] J. Peters, S. Vijayakumar, and S. Schaal, "Natural actor-critic," in *Machine Learning: ECML 2005: 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005. Proceedings 16*, pp. 280–291, Springer, 2005.

[6] T. Degris, M. White, and R. S. Sutton, "Off-policy actor-critic," *arXiv preprint arXiv:1205.4839*, 2012.

[7] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*, pp. 1889–1897, PMLR, 2015.

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

[9] F. Pardo, A. Tavakoli, V. Levdik, and P. Kormushev, "Time limits in reinforcement learning," in *International Conference on Machine Learning*, pp. 4045–4054, PMLR, 2018.

[10] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*, pp. 1861–1870, PMLR, 2018.

[11] 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.

[12] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Conference on Robot Learning*, pp. 91–100, PMLR, 2022.

[13] L. Schneider, J. Frey, T. Miki, and M. Hutter, "Learning risk-aware quadrupedal locomotion using distributional reinforcement learning," *arXiv preprint arXiv:2309.14246*, 2023.

[14] "Mujoco 3."

[15] T. Miki, J. Lee, L. Wellhausen, and M. Hutter, "Learning to walk in confined spaces using 3d representation," 2024.

[16] F. Ruscelli, A. Laurenzi, N. G. Tsagarakis, and E. M. Hoffman, "Horizon: a trajectory optimization framework for robotic systems," *Frontiers in Robotics and AI*, vol. 9, 2022.

[17] B. Dynamics, "Stepping Up, Reinforcement Learning with Spot." `https://www.youtube.com/watch?v=Kf9WDqYKYQQ`, 2023. [Online; accessed 23-March.-2024].

[18] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg, "Orbit: A unified simulation framework for interactive robot learning environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3740–3747, 2023.

[19] T. Howell, N. Gileadi, S. Tunyasuvunakool, K. Zakka, T. Erez, and Y. Tassa, "Predictive Sampling: Real-time Behaviour Synthesis with MuJoCo," dec 2022.

[20] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, "Learning-based model predictive control: Toward safe learning in control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, pp. 269–296, 2020.

[21] A. Patrizi, "LRHControl." `https://github.com/AndrePatri/LRHControl`, 2023. [Online; accessed 10-March.-2024].

[22] A. Patrizi, "OmniRoboGym." `https://github.com/AndrePatri/OmniRoboGym`, 2023. [Online; accessed 10-March.-2024].

[23] A. Patrizi, "CoClusterBridge." `https://github.com/AndrePatri/CoClusterBridge`, 2023. [Online; accessed 10-March.-2024].

[24] A. Patrizi, "RHCViz." `https://github.com/AndrePatri/RHCViz`, 2023. [Online; accessed 10-March.-2024].

[25] A. Patrizi, "SharsorIPCpp." `https://github.com/AndrePatri/SharsorIPCpp`, 2023. [Online; accessed 10-March.-2024].