IBRIDO: A Software Ecosystem for Research in Reinforcement Learning-based Receding Horizon Control[§]

Andrea Patrizi, Carlo Rizzardo and Nikos G. Tsagarakis

Università di Genova

iit HHCM ISTITUTO ITALIANO DI TECNOLOGIA HUMANOIDS AND HUMAN CENTERED MECHATRONICS

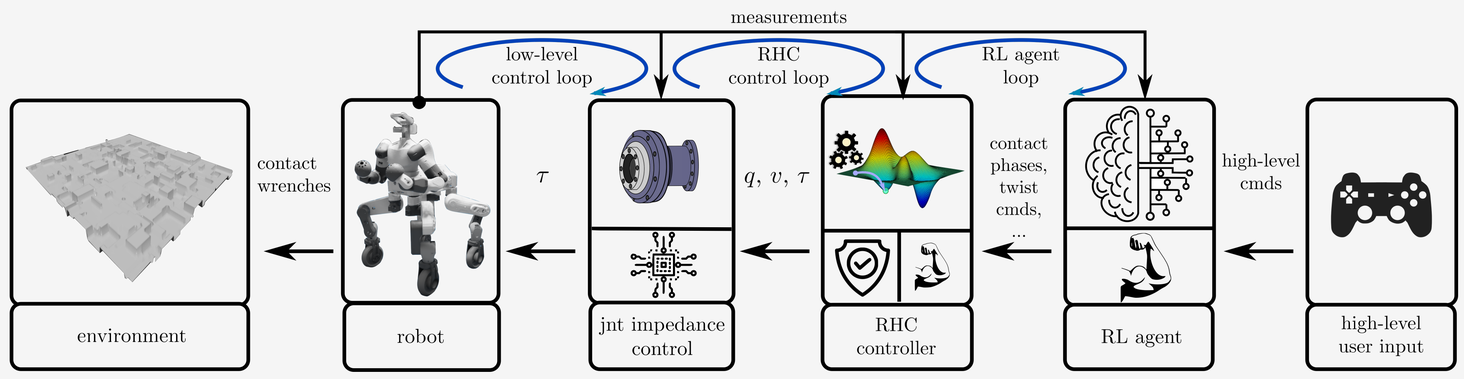
[§] This project has received funding from the European Union's Horizon Europe Framework Programme under grant agreement No 101070596

A Hybrid Approach: combining Receding Horizon Control and Reinforcement Learning

Most popular approaches:

- ▷ *Model augmentation*: improving prediction accuracy and control performance
- ▷ *Adaptive tuning* and *parameter optimization*: parameters tuning (e.g. weights, costs, constraints)
- ▷ *Safety*: learned-policy coupled with RHC controller, *safety filter*

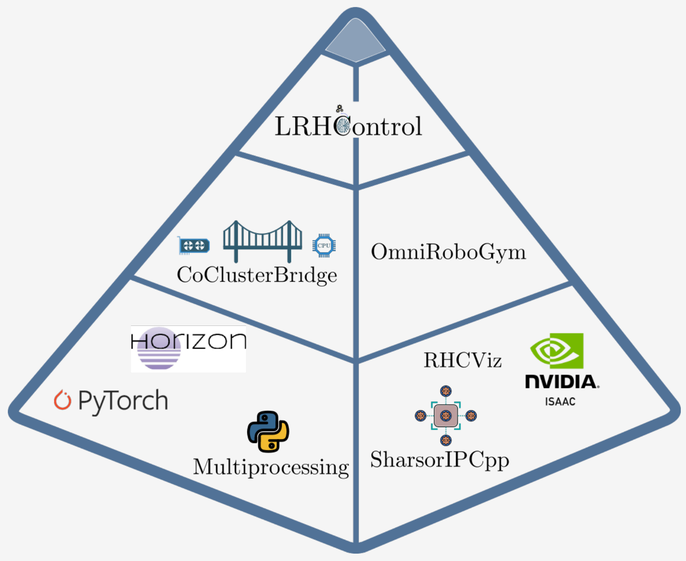
Our approach:



- ▷ *Hierarchical coupling*: agent on top of RHC controller
- ▷ RHC used as *local* and *safety-aware* controller
- ▷ Agent for high-level task achievement and RHC *performance* exploitation

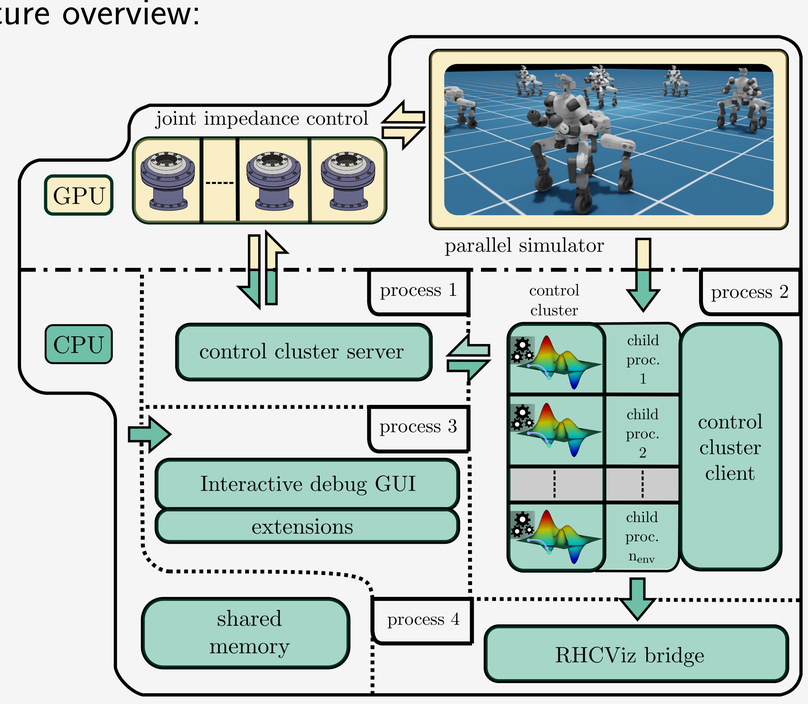
Software architecture

Software hierarchy:



- ▷ *LRHControl* → main package; sim, train env, task implementations.
- ▷ *CoClusterBridge* → coordinates connection between simulator and cluster of RHCs
- ▷ *OmniRoboGym* → wrapper on top of IsaacSim; base sim env
- ▷ *SharsorIPCpp* → shared memory backend
- ▷ Horizon (iLQR solver) → formulation of RHC controllers (CPU)
- ▷ Multiprocessing → RHCs parallelization on CPU
- ▷ GPU-accelerated simulation for data collection (IsaacSim)
- ▷ PyTorch for DL components

Architecture overview:

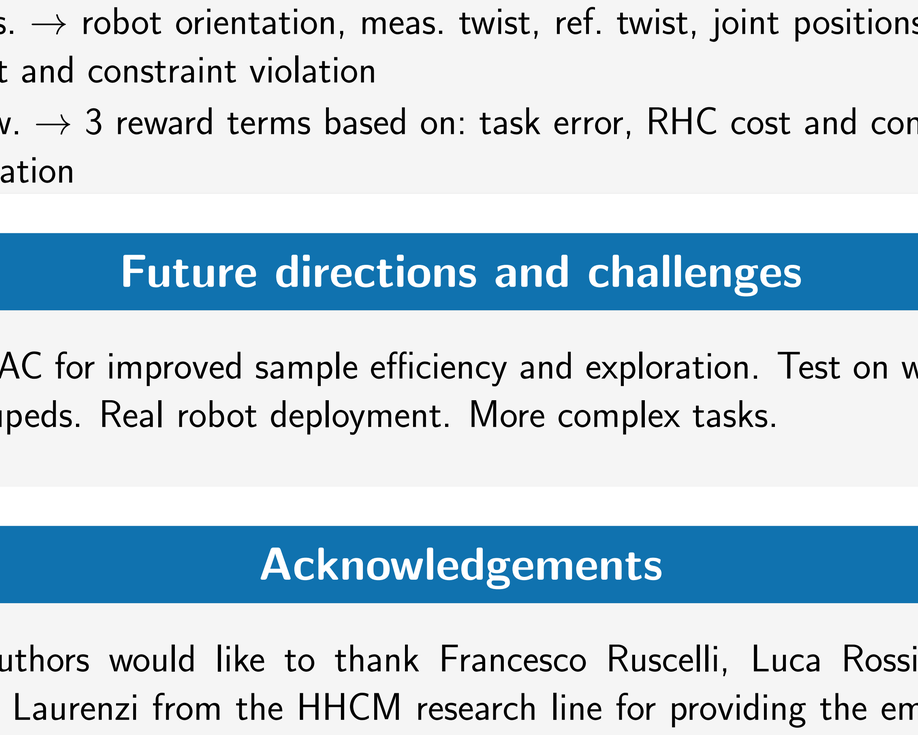


- ▷ Separate simulation and training env
- ▷ RHCs run in parallel, synchronously w.r.t. simulator
- ▷ Simulation stepping also parallel w.r.t. cluster solution
- ▷ Extensible debugging GUI for monitoring RHCs, training and for easier RL task tuning
- ▷ Single env visualization thanks to *RHCViz*

A proof-of-concept example: preliminary results

Workstation configuration used for testing:

- ▷ 32 core 13th Gen Intel(R) Core(TM) i9 - 13900KS workstation, 64 GB RAM, NVIDIA RTC 6000 Ada Generation
- ▷ Roughly 30 GB RAM necessary for running 256 RHCs + simulator (256 envs). 25% GPU usage at runtime, roughly 7 GB/48 GB



Proof of concept task:

- ▷ PPO algorithm for training
- ▷ Custom RHC controller for quadruped locomotion (closed source)
- ▷ Locomotion on flat ground exploiting RHC
- ▷ Act. → agent can command the RHC through twist commands and contact phases selection
- ▷ Obs. → robot orientation, meas. twist, ref. twist, joint positions, RHC cost and constraint violation
- ▷ Rew. → 3 reward terms based on: task error, RHC cost and constraint violation

Future directions and challenges

Test SAC for improved sample efficiency and exploration. Test on wheeled quadrupeds. Real robot deployment. More complex tasks.

Acknowledgements

The authors would like to thank Francesco Ruscelli, Luca Rossini and Arturo Laurenzi from the HHCM research line for providing the employed RHC controller and the technical support throughout the project.

Resources and contact information

Data ↓

https://github.com/AndrePatri/IBRIDO

andrea.patrizi@iit.it;
carlo.rizzardo@iit.it