Expert-Guided Imitation for Learning Humanoid Loco-Manipulation from Motion Capture

Rohan P. Singh[1†], Pierre-Alexandre Leziart[1†], Masaki Murooka[1], Mitsuharu Morisawa[1], Eiichi Yoshida[2], Fumio Kanehiro[1]

***Abstract*— Despite significant advances in bipedal locomotion, enabling humanoid robots to perform general whole-body tasks through meaningful interaction with their environments remains a challenging open problem. While deep reinforcement learning (RL) has recently demonstrated impressive results in dynamic walking — even on complex and unpredictable terrain — real-world utility demands that humanoids go beyond locomotion to execute task-oriented behaviors.**

In this work, we propose a framework for teaching humanoid robots to imitate humans doing useful tasks by training policies for tracking human motion references. Our approach leverages high-quality in-house motion capture (MoCap) data, from which we perform kinematic retargeting to project human trajectories onto a humanoid platform. Crucially, we adopt a hybrid learning paradigm: the policy is trained to track upper-body and root motions from the MoCap data, and receives additional supervision from a pre-trained omnidirectional walking expert. This expert guidance, implemented via a Behavior Cloning (BC) objective, ensures that leg motion respects dynamics and kinematic constraints of the humanoid. We train policies entirely in simulation and successfully transfer them to a real humanoid robot. We validate our method on a box loco-manipulation task, demonstrating effective sim-to-real transfer and marking a step toward more capable, task-driven humanoid behavior.

I. Introduction

We propose a two-stages approach for humanoid loco-manipulation that **is implemented on top of a publicly available framework for reproducibility purpose** [1].

In summary, our contributions are the following:

1) We design a two-stages learning pipeline for humanoid box loco-manipulation that relies on a **mix of reinforcement learning and behavior cloning** with Proximal Policy Optimization (PPO) [2] and Adversarial Motion Priors (AMP) [3] for frame-by-frame motion tracking.
2) We study the robustness and generalization of the trained policy under variability between the data capture scene and the policy deployment scene.
3) We highlight how this generalization can be exploited for **long distance transport** by manipulating the pick and drop locations.
4) We validate our framework by **deploying it to a real-world H1 humanoid robot**, as seen in Figure 1, and demonstrating the reproducibility of the whole pick-and-drop cycle.

† Equal contribution.

[1] CNRS-AIST JRL (Joint Robotics Laboratory) IRL, National Institute of Advanced Industrial Science and Technology (AIST), Japan.

[2] Tokyo University of Science, Tokyo, Japan.

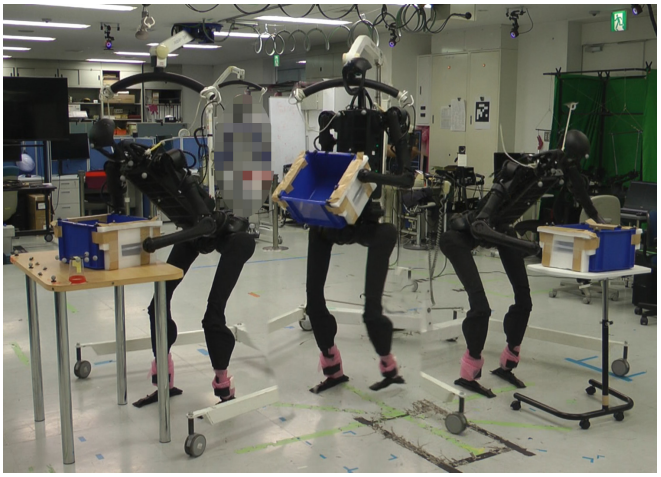


Fig. 1: Loco-manipulation scenario with a H1 humanoid robot: picking a box up and dropping it down on another table.

II. Approach

Our objective is to develop a methodology that enables humanoid robots to learn loco-manipulation behaviors directly from human motion demonstrations.

• **Motion capture recording:** The pipeline begins with the collection of reference motion data in-house by having a human subject perform the loco-manipulation task while wearing a full-body motion capture suit. This includes walking towards a box placed on a table, picking it up, then walking to another table to drop it down. We record both the 3D trajectories of body markers and the corresponding skeletal motion. We also record contact forces through sensors placed on the hands and on the bottom surface of the box for accurate identification of contact events during interaction.

• **Motion retargeting with IK:** The kinematic retargeting of the collected MoCap trajectory from a human skeleton to a humanoid robot is formulated as an optimization-based inverse kinematics through a constrained quadratic programming problem. The objective function minimizes the weighted sum of squared errors between the 3D positions and orientations of key end-effectors (hands and feet) as well as the pose of the torso, and the head. This formulation allows for smooth tracking of human motion while preserving the structural characteristics of the original trajectory.

• **Motion tracking with RL:** The retargeted reference data we obtain allows us to use RL to train policies in simulation for imitating the human motion. **Observations** include the usual proprioceptive measurements and the relative pose of the target scene element. We also introduce

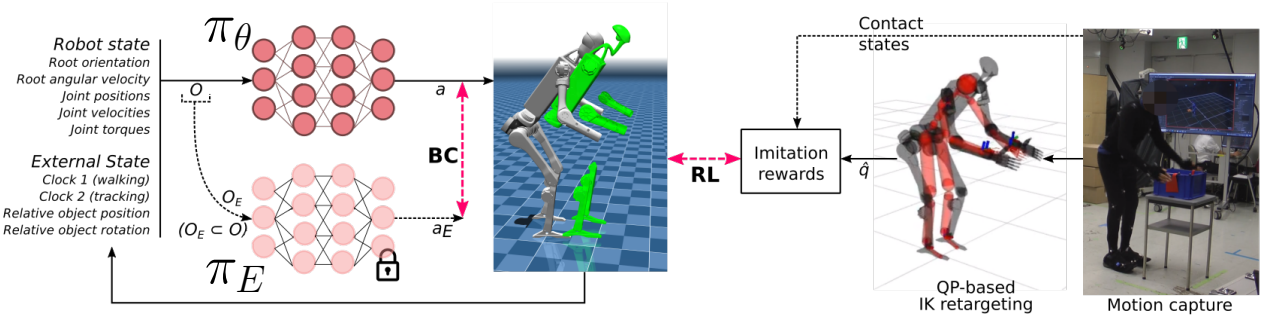


Fig. 2: **Overview of the proposed hybrid training approach.** The whole-body imitation policy π_θ is rewarded to track only the upper body and root motion of the human demonstrations while an auxiliary behavior cloning loss term provides supervision from an expert walking policy π_E. The student policy learns to control all joints of the robot and successfully achieves the loco-manipulation task.

two clock signals: a phase variable for synchronizing the robot with the reference state, and a periodic clock signal for bipedal gait, which we found to be helpful with sim-to-real transfers especially for networks without an observation history. **Rewards** includes terms for tracking the (1) root height and orientation, (2) relative poses of torso, left arm, and right arm in pelvis frame, (3) joint position and velocity (4) relative pose of the target in pelvis frame, (5) contacts, and (6) relative positions of the hands in the frame of the box. We also include regularization terms for minimizing joint torques, and penalties for joints near the range limits.

- **Decoupled supervision for loco-manipulation:** While our motion tracking reinforcement learning framework enables the robot to learn to imitate human motion, directly mimicking full-body trajectories including walking movements is infeasible due to the substantial dynamic and morphological differences between humans and humanoid robots. In particular, the discrepancy in limb proportions and joint constraints often leads to violations of dynamic stability when attempting to directly replicate human walking patterns. Our early sim-to-real experiments showed that the robot struggles to make stable foot contacts with the floor while making unrealistically long strides. Furthermore, tuning IK parameters to enforce both kinematic accuracy and plausible contact dynamics such as maintaining foot orientation is labor-intensive and not scalable.

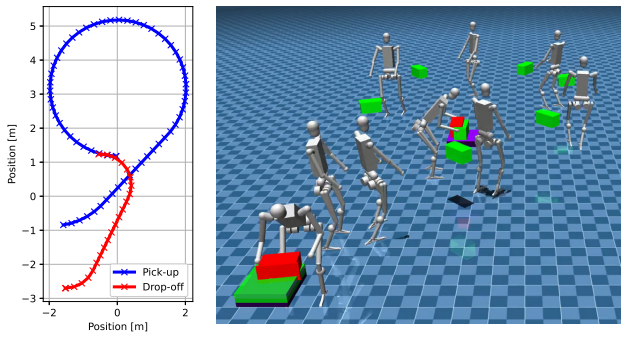
Thus we train an expert bipedal walking policy by adapting for H1 the approach presented in [4], [5]. We implement a teacher-student supervision through a behavior cloning objective, guiding the student policy to match the expert's leg actions. This hybrid approach allows the student policy to benefit from high-level human demonstrations while leveraging the robustness and stability of the expert locomotion policy, resulting in coherent whole-body behavior that successfully integrates walking and manipulation.

We train 3 policies respectively for pick-up, drop-off and return to the origin. Figure 2 highlights the whole hybrid training architecture.

III. Simulation and Experimental Results

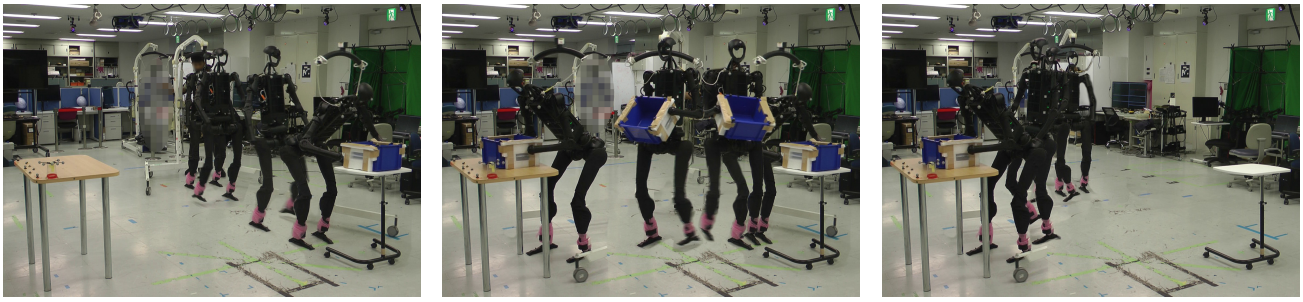
This pipeline relies on the open-source CPU-based simulation engine Mujoco [6] combined with Ray [7] as a parallelization framework to scale training on several cores. First, we leverage the PPO algorithm [2] to train the expert walking policy on flat ground for a velocity tracking task. We apply random pushes to the robot, dynamics randomization and sensor noise, and we add random bumps on the ground as additional disturbances. The imitation policy is then trained with all domain randomization disabled. It is further finetuned after convergence for a few thousands epochs by re-enabling all randomization for better sim-to-real transfer. The whole process amounts to around 36 hours of training using a 32-cores AMD Threadripper PRO 5975WX to gather samples from 32 environments simultaneously.

- **Robustness to target offsets:** For potential future applications, we study the success rate of the policy for various pick-and-drop positions and orientations around the baseline training scenario. The robot achieves a consistent pick-up of the box over a wide range of offsets roughly between $[-2.0, 0.4]m$ and $[-0.8, 0.8]m$ for the longitudinal and lateral axes, $[-5, 15]cm$ in height and $[-0.4, 0.4]rad$ in orientation. This highlights the efficiency of domain randomization for handling setups that deviate from the motion capture recording, thus extending the robot workspace.
- **Extension to long distance loco-manipulation:** This capacity to handle a wide range of deviations from the reference demonstration pushed us into exploring loco-manipulation over long distances. To do so, we leverage a 2D planning algorithm to generate paths along which fake



(a) Dubins paths. (b) Snapshots of pick-up motion and final drop-off.

Fig. 3: Dubins path for the pick and drop motion (left) and snapshots along the trajectory (right). Intermediate frames of the drop-off part are omitted for clarity. The fake box and table positions are displayed in green. Due to the box position and the path curvature constraint, the robot does not directly move to the box but instead moves past it before looping back.



(a) Picking up the box. (b) Dropping off the box. (c) Going back to starting position.

Fig. 4: Full loco-manipulation cycle by switching policies online. The support crane has been partially edited out for visibility purpose.

target positions will be set to sequentially lead the robot to the real pick-up and drop-off locations. We integrate in our pipeline the open-source RRT-Dubins path planner [8]. Figure 3 highlights a pick-and-drop sequence that illustrates a generalized use-case of the policy that goes beyond the single motion capture recording we performed.

• **Real-world deployment:** After training in simulation, the controller is directly deployed on a real Unitree H1 robot [9]. The policy runs at 40 Hz on a standard laptop computer by using the Open Neural Network Exchange (ONNX) framework [10] through the ONNX Runtime inference engine [11]. Communications with the robot are ensured by the Unitree SDK2 [12] through an Ethernet connection. As the focus of this work is not to perform fully autonomous demonstrations, we simplify the experimental setup by using motion capture instead of onboard sensors to track the position of the robot, the box and the drop-off table. We only roughly place them with respect to each other so small position and orientation offsets are to be expected compared to the nominal training setup.

Real-world motion control results are shown in Figure 4. The robot achieves a full loco-manipulation cycle by walking toward the box, picking it up to drop it off on another table, then going back to its starting position. This is done in one go by automatically switching between policies when the phase clock signal reaches its final value. The cycle can be repeated on-the-fly by placing the box back on the first table while the robot is returning to the starting position. This successful deployment indicates that the domain randomization we used was effective for crossing the sim-to-real gap.

While the locomotion part of the motion is robust and repeatable, most failures occur when picking-up the box. Failures during the drop-off sequences mostly occurred when the box was not properly picked-up, with the box falling from the robot's hand before reaching the table. The differences in dynamics for the contact interactions between our training setup and reality partially explain these failures. Moreover, since H1 only has relative encoders in his arms, slight calibration errors lead to joint position offsets, which can worsen the quality of the pick-up motion. Encouraging the robot to apply more forces on the box during training could be a way to address this issue as the real robot would then probably bring its hands closer. However, this is not an ideal solution for general loco-manipulation as fragile or soft packages might be damaged by such an approach.

Future improvements could come from a better contact perception by the policy so that the robot can react online and correct improper pick-ups.

ACKNOWLEDGEMENTS

The authors thank all members of JRL for providing their support in conducting robot experiments that were done during the production of this work. This work was partially supported by JSPS KAKENHI Scientific Research (S) Grants Number JP22H05002 and JP24KF0125, and by the Japan Society for the Promotion of Science (JSPS) Postdoctoral Fellowships for Research in Japan.

REFERENCES

[1] R. P. Singh, "Learning humanoid walking," 2025. [Online]. Available: https://github.com/rohanpsingh/LearningHumanoidWalking

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

[3] 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.

[4] R. P. singh, Z. Xie, P. Gergondet, and F. Kanehiro, "Learning bipedal walking for humanoids with current feedback," *IEEE Access*, vol. 11, pp. 82 013–82 023, 2023.

[5] R. P. Singh, M. Morisawa, M. Benallegue, Z. Xie, and F. Kanehiro, "Robust humanoid walking on compliant and uneven terrain with deep reinforcement learning," in *2024 IEEE-RAS 23rd International Conference on Humanoid Robots (Humanoids)*. IEEE, 2024, pp. 497–504.

[6] 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*. IEEE, 2012, pp. 5026–5033.

[7] P. Moritz, R. Nishihara, S. Wang, A. Tumanov, R. Liaw, E. Liang, M. Elibol, Z. Yang, W. Paul, M. I. Jordan, *et al.*, "Ray: A distributed framework for emerging {AI} applications," in *13th USENIX symposium on operating systems design and implementation (OSDI 18)*, 2018, pp. 561–577.

[8] F. Cantalloube, "Rrt-dubins," 2025. [Online]. Available: https://github.com/FelicienC/RRT-Dubins

[9] Unitree. (2025) Unitree products. [Online]. Available: https://shop.unitree.com/products

[10] O. developers, "Onnx," 2018. [Online]. Available: https://github.com/onnx/onnx

[11] O. R. developers, "Onnx runtime," 2018. [Online]. Available: https://github.com/microsoft/onnxruntime

[12] Unitree, "Unitree sdk 2," 2024. [Online]. Available: https://github.com/unitreerobotics/unitree_sdk2