Less Reward is More: Improved Reinforcement Learning Control of a Mobile Manipulator using Clamped Joints

Denis Hadjivelichkov DENNIS.HADJIVELICHKOV@UCL.AC.UK
University College London (UCL)

Kostas Vlachos KOSTASWL@CSE.UOI.GR
Univeristy of Ioannina

Dimitrios Kanoulas D.KANOULAS@UCL.AC.UK
University College London (UCL)

Abstract

Many robotic path planning problems are continuous, stochastic, and high-dimensional. The ability of a mobile manipulator to coordinate its base and manipulator in order to control its whole-body online is particularly challenging when self and environment collision avoidance is required. Reinforcement Learning techniques have the potential to solve such problems through their ability to generalise over environments. We study joint penalties and joint limits of a state-of-the-art mobile manipulator whole-body controller that uses LIDAR sensing for obstacle collision avoidance. We propose directions to improve the reinforcement learning method. Our agent achieves significantly higher success rates than the baseline in a goal-reaching environment and it can solve environments that require coordinated whole-body control which the baseline fails.

1. INTRODUCTION

Mobile robots have a plethora of applications ranging from warehouse services, through oil rig inspections, to emergency interventions [Rajana (2018); Bengel et al. (2009); Rehak et al. (2013)]. Modern robots require both high mobility and accurate manipulation to traverse collision-free paths while performing their tasks, which can be achieved via *mobile manipulators*. By applying Whole-Body Control (WBC), the base and manipulator movements of mobile robots coordinate to improve the efficiency of the system.

Classical WBC methods include the use of kinematic, velocity, and impedance controllers, model predictive controllers, and combinations thereof in advanced adaptive control strategies [Moro and Sentis (2018); Logothetis et al. (2018); Kim et al. (2019)]. They have been shown to work well in many environments while also providing stability guarantees. Reinforcement Learning (RL) methods have shown great promise in their ability to compete with and potentially overcome classical methods in many robotic problems as they can work with complex inputs [Jaderberg et al. (2017)] and learn complex task solutions [Levine (2018)]. Once trained, RL agents can execute policies online, bringing down total mission times. The recent state-of-the-art works on RL for mobile manipulator WBC have focused on goal reaching scenarios. While they show that their solutions are quicker than traditional methods they still underperform them in terms of success rate and impose big limitations on the robots and their environments, such as limited DoF [Kindle et al. (2020)] and simplistic tasks [Wang et al. (2020)]. In this paper, we examine the trained

WBC behaviour of a state-of-the-art baseline agent trained with a shaped reward. The robot agent is comprised of an omnidirectional base and a 7 DoF manipulator simulated in PyBullet. We determine potential causes of sub-optimality in the baseline - frequent early episode termination due to reaching joint limits and consistently folded arm. We show that clamping the joint instead of a joint-limit penalty in the the reward improves the model's performance significantly and allows it to reach the goal much closer. The summarised contributions of this paper are: (i) Identifying issues and potential improvements of a state-of-the-art method; (ii) Showing our method leads to higher success rates than the baseline and solves an environment requiring whole-body control, which the baseline fails; (iii) Evince our method's ability to generalise in an unseen environment.

2. RELATED WORK AND BACKGROUND

In this section, we present traditional and reinforcement learning approaches to whole-body control. Then, we provide the baseline reinforcement learning background setup.

Traditional Approaches: Traditional approaches to implementing WBC include the use of kinematic and dynamic controllers [Khatib (1987); Wu et al. (2019)]. Their advantage is that the current understanding of physical systems is refined and works well on fully actuated robots. Most methods focus on WBC for multi-legged robots [Dietrich et al. (2012); Hoffman et al. (2018); Rolley-Parnell et al. (2018); Laurenzi et al. (2019)]. Model Predictive Control methods are popular with works such as Minniti et al. [Minniti et al. (2019)] showing success in WBC pose-tracking and interaction tasks. Recent works focus on non-linear strategies, such as non-linear model predictive control [Logothetis et al. (2018)]. While some methods such as Operational Space Control can solve tasks with optimality and continuity in real-time, most traditional methods require large offline computation. Moreover, the methods for mobile manipulators are often based on simplified models of the robot which sometimes results in control solutions that are limiting the its agility.

Reinforcement Learning Approaches: Reinforcement learning approaches offer a framework that is transferable to different tasks and robots, able to work online with scaling complexity, in a trade off with the limited prior information that it can use and long training that is often required for good performance. However, current methods still use application-specific architectures and rarely generalize to multi-task scenarios [Sammut (2012)]. RL methods have successfully taught robots dexterous vision-based manipulation tasks [Julian et al. (2020)] and navigation tasks Hester et al. (2012); Francis et al. (2019). Most research is also focused on legged robots [Li et al. (2018); Lober et al. (2016)]. Wang et al. [Wang et al. (2020)] integrate the state-of-the-art RL algorithms with visual perception for WBC and propose an efficient framework for decoupling of visual perception from control, which enables easier sim-to-real transfer However, the used environment is simple, consisting of a table in front of a robot. Kindle et al. [Kindle et al. (2020)] use a Proximal Policy Optimization (PPO) based agent to train end-to-end whole-body control policies for obstacle avoidance and tested on a real mobile manipulator achieving state-of-the-art results. Their model makes use of Automatic Domain Randomization and Continuous Learning to guide the agent toward a solution in a custom reach-and-grasp environment. A hand-crafted reward function is defined with components for collision, joint limits, safety distance, optimal path following and time. These recent works show sub-optimal performance, worse than traditional methods.

2.1. Baseline Background

We consider a standard RL framework, which includes an agent interacting with an environment via actions and observations. Environment rewards are fed into an RL learning algorithm, which optimises the agent's policy and thus creates a feedback loop. The problem focuses on goal-reaching environments in which a success is defined as the uninterrupted holding of the robot agent's end-effector within a given tolerance distance from the goal. The environments' state space consists of front and rear LIDAR scans, arm joint positions, arm joint and base velocities, and the goal location in the end-effector frame, while action space consists of joint and base accelerations. Both LIDAR observations and joint actions are limited to 2D planes. In this section, we discuss he state-of-the-art baseline [Kindle et al. (2020)] used for our experiments.

Reward: The baseline's reward function is handcrafted, encouraging the agent to learn to imitate a traditional path planning method and complete the task quicker, while discouraging it for moving close to objects. The reward has three termination cases: collision, timeout, and reaching joint limits. Finally, it also introduces an accumulation term that prevents the agent's exploitation of the reward.

Agent: The architecture of the agent is based on PPO with modified layers as depicted in Fig. 1. The two LIDAR scans are compressed via a separate scan block before being processed with the rest of the inputs in a network of fully connected layers. The agent produces a discretized policy for each action.

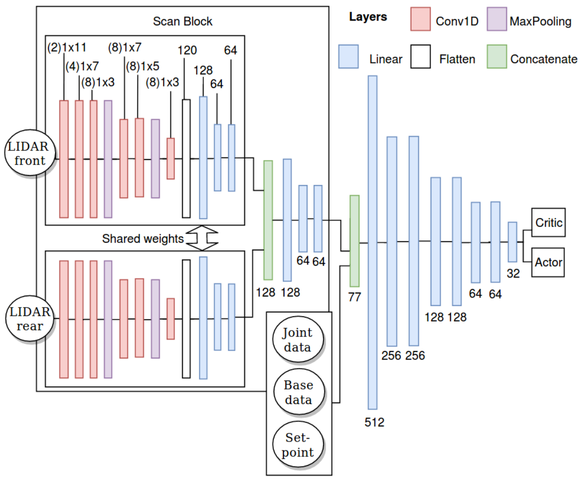

Figure 1: Agent's network architecture.

3. METHOD

To understand the low success rate of the baseline in comparison with traditional methods, we analysed the behaviour and performance of the agent after training. It was observed that the majority of episodes terminate due to the robot arm reaching its joint limits. We further noticed that this is a behaviour that can be limited explicitly instead of penalising and terminating the reinforcement learning agent. While it was expected that the optimal solution would be for the robot to move toward the goal with a folded arm and unfold it while it is reaching the goal position, it was observed that the robot folds the arm in the beginning and does not change it throughout the whole run, as shown in Fig. 2 as well as the real robot experiments [ASLteam]. The cause of this could be partially explained by the custom environment itself, which does not explicitly require WBC in order to be solved. Additionally, the used reward function itself places more weight on optimal path following penalties than on timing penalties - following the optimal end-effector path is simpler when the manipulator is folded, because the end-effector is close to the base point of rotation. We attempt to address some of these drawbacks in this section.

Improved Environments: Two environments are used in our validations: a narrow corridor environment for comparison with state-of-the art and a new environment that cannot be solved without WBC. The first environment, adapted from the baseline, consists of a narrow corridor of

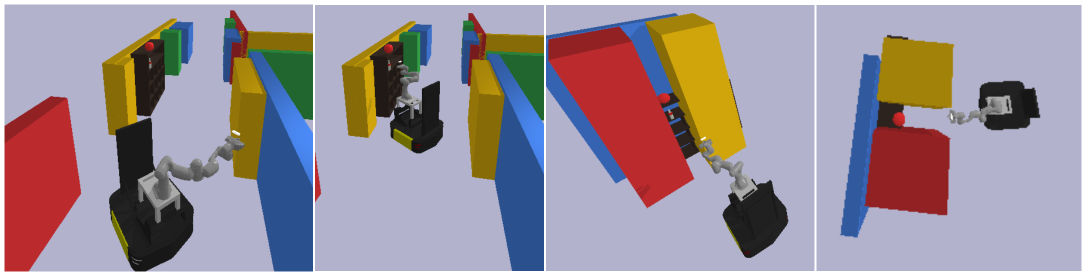

Figure 2: The *Corridor-env* (left two) requires path planning to be solved, but allows for the base and manipulator to move independently. The local *Gap-env* (right two) cannot be solved without coordination. Goal represented by red sphere.

variable length, containing random avoidable obstacles and a randomly placed goal location (See Fig. 2). It requires the agent to plan its path and navigate around the obstacles toward the goal. We refer to this environment as *Corridor-env*. However, its goal can be directly reached by folding the arm and performing only mobile base collision-free navigation, thus does not require WBC.

We introduce a new environment (referred as *Gap-env*) which consists of narrow passages with the goal end-effector pose being reachable only with coordination between the base and manipulator (See Fig. 2). The width of the gap is very narrow and a small deviation of the arm or base during insertion would cause a collision, thus the task is difficult to solve without coordinated control. Two variants are used: In *Gap-env-train*, random uniform noise is added to initial joint angles, and orientation and position of the goal relative to the robot spawn location; In *Gap-env-test*, the tunnel gap width and length, as well as the goal placement relative to the tunnel is also randomly initialized to ensure that the testing scenarios are unseen by the agent.

In all the environments, a success is defined as the uninterrupted holding of the robot agent's end-effector within a given tolerance distance from the goal. Automatic Domain Randomization (ADR) is used to gradually adapt the complexity of the environment and guide the agent toward a solution. This is done by increasing or decreasing the acceptable tolerance distance to the goal depending on the agent's recent success rate. Via ADR, the tolerance distance to the goal is dynamically changed. The state space consists of 2D front and rear LIDAR scans, arm joint positions, arm joint and base velocities, and the goal location in end-effector frame. The action space is comprised of mobile base and arm joint accelerations. Given the complexity of the problem, its dimensionality is reduced to planar movements of the arm.

Joint Clamping Method: When training the baseline agent in the corridor environment, it was noticed that most of the episodes end due to joint limit termination. However, joint limits can easily be enforced by setting a manual limit (clamping) to the joint positions based on the robot's hardware limits with appropriate tolerance to protect the robot. Likewise, when training the baseline in the gap environment, it doesn't approach the goal, rather stays near the gap and oscillates. This is believed to be due to the baseline's safety margin penalty, which encourages the robot to keep its distance from all objects. We believe that a collision termination penalty is sufficient in teaching that behaviour. Thus, our modified reward function is as follows:

$$
\begin{aligned}
r_t = w_t \cdot \frac{\tau}{T_t} + w_{pd}\Delta d_{pd} + w_{pt}\frac{\Delta d_{pt}}{d_{pt,init}} + w_{ht}\frac{\tau}{T_h} \\
+ w_{hd}(1 - \min(1, d_g/d_h))\frac{\tau}{T_h} - I_h + D_c + D_h
\end{aligned}
\tag{1}
$$

where w_t is the time penalty parameter, τ is the step time, and T_t is the total time before episode timeout. This timeout reward encourages quicker task completion. An optimal path towards the goal is computed via Harmonic Potential Field (HPT). The goal distance reward penalises for deviation from the HPT Δd_{pd} by w_{pd} and rewards movement along the path Δd_{pt} normalized by the total path $d_{pt,init}$ by w_{pt}. Furthermore, w_{ht} is the reward for each time-step that the end-effector is within tolerance distance d_h of the goal point and w_{hd} is the reward for minimizing the distance to the goal position applied only when the distance to the goal d_g is smaller than the tolerance distance d_h. Both of these rewards are normalized for the holding time threshold T_h after which the task is done. I_h is the accumulated holding reward which is subtracted if the end-effector leaves the tolerance sphere in order to prevent exploitation of the reward. Finally, D_c is a collision penalty, and D_h is the reward for sustained holding time T_h. The last two rewards end the current episode.

This reward function allows the agent to actuate the robot safely without hindering its learning and addresses the two issues encountered when running the baseline. The joint clamping is enforced programmatically, based on the robot's joint limits. The agent is trained with these modifications and compared with the standard baseline for several values of goal tolerance distance in Sec. 4.1.

Validation Setup We use a mobile manipulator robot comprised of a omnidirectional base and a 7 DoF arm manipulator. All simulations are done using PyBullet 2.8 [Erwin Coumans (2016–2020)], while a high performance computing cluster is used for training. Agent parameters are shown in the appendix. Our method and the baseline are trained in *Corridor-env* on 32 parallel workers for a total of 60M training steps. The final success rate, counted as number of successes over 100 episodes, is compared in Sec. 4.1. We train the agents in *Gap-env-train* for 30M steps with 16 workers. The agent is then evaluated in *Gap-env-test* and the results are reported in Sec. 4.2. In both environments, the ADR is gradually adapting the tolerance distance in the range $[0.5; 0.05]$.

4. RESULTS

We explore the following questions: (i) How does our method compare to the baseline? (ii) Is the new agent able to perform well on a task requiring Whole-Body Control? (iii) Is the agent generalizable to new environments?

4.1. Validation on Corridor Environment

	Tolerance Dist (m)			
	0.5	0.2	0.1	0.07
Baseline	72%	65%	40%	0%
Joint-Clamping (ours)	**73%**	**73%**	**63%**	**24%**

Environment	
Gap-env-train	Gap-env-test
0%	0%
81%	76%

Table 1: Success rates in (left) *Corridor-env* against tolerance distances and in (right) *Gap-env* with $0.05m$ tolerance distance.

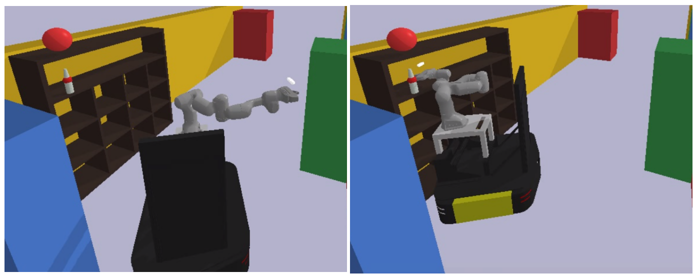

Figure 3: Our agent at start (left) and end (right) of reaching task.

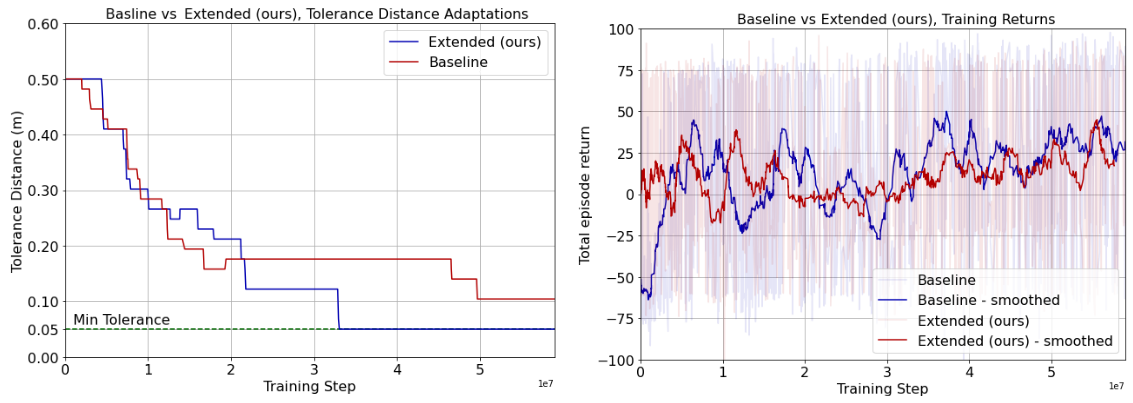

Figure 4: **(left)** Adaptation of tolerance distance per step for both reward settings. **(right)** Total returns per episode plotted against the episode termination step. Running mean smoothing of 0.95 is used. Note that due to the use of ADR, the training rewards are fairly similar

The original baseline agent with a joint limit penalty and our modified agent with clamped joint limits (Eq. 1) are ran with 32 workers for 60M steps. This process took 48 hours to finish. The trained models were tested in *Corridor-env* with fixed goal tolerance distances in the range of 0.5 to 0.07. Running the baseline, we managed to achieve 72% for the highest tolerance distance (0.5m), while the success rate was significantly decreasing with the tolerance distance dropping. The minimum successful tolerance distance was 0.1m. The resulting success rates of our agent, shown in the lower row of Table 1, are noticeably higher than the baseline performance with standard reward, especially when the tolerance distance is decreasing.

This difference in performance can be further explained by the difference of ADR tolerances shown in Fig. 4. For the baseline, the ADR tolerance distance reached at 60M steps is 0.1, not reaching the lowest distance of 0.05. In comparison, our agent successfully adapted to the lowest tolerance distance 20M steps before the training ended. This indicates that with the modified reward, the agent learns quicker and better. The training returns of the original and modified baselines are shown in Fig. 4. For the modified agent in the environment with tolerance distance fixed to $0.07m$, it is found that 52% of the unsuccessful episodes terminate due to collision, while 48% terminate due to timeout. The distance from the end-effector to the goal at the end of unsuccessful episodes is on average $0.11m$. While the joint limit modification shows an increase in success rate, the folded manipulator behaviour is still observed (see Fig. 3).

4.2. Whole-Body Control Task

Training the agent locally in *Gap-env*, which includes narrow tunnels where only the arm can fit, forces the simultaneous coordination between the arm and the mobile base as a WBC. With a 0.05m tolerance distance to the goal, the success rate of the training is 81%, shown in Table 1. As can be observed from the success rates in the table, our agent successfully generalises to unseen variants of the training environment, with a drop of only 5% in success rate. In a typical episode, the robot is observed moving towards the goal, while adjusting its manipulator for tunnel-entry, as expected from a WBC solution. In failed episodes, it is seen that the robot often reaches the goal within less than $0.05m$, however it backs off and re-approaches several times until the episode terminates due to timeout. Note that the original baseline method was not able to solve such environments.

5. CONCLUSIONS AND FUTURE WORK

This work presents an RL method for Whole-body Control of a mobile manipulator that improves on the state of the art. Our shaped reward function combined with joint limit clamping shows a significant improvement of 24% over the baseline for small tolerance distances. Moreover, the proposed agent can solve WBC tasks which the baseline fails. We show that training with our reward in one environment, transfers its learned skills well to a similar, but different, testing environment. The current method is limited to using 2D LIDAR data and planar manipulator actions. Future work will focus on expanding these limitations via more informative observations, such as 3D LIDAR or RGB-D images. More importantly, the "folding arm" behaviour should be further examined.

Acknowledgment: We thank Julien Kindle for his support in running the baseline agent code.

References

ASLteam. Whole-Body Control of a Mobile Manipulator using End-to-End Reinforcement Learning. https://www.youtube.com/watch?v=3qobNCMUMV4.

Matthias Bengel, Kai Pfeiffer, Birgit Graf, Alexander Bubeck, and Alexander Verl. Mobile Robots for Offshore Inspection and Manipulation. In *IEEE/RSJ IROS*, pages 3317–3322, 12 2009.

Alexander Dietrich et al. Reactive Whole-Body Control: Dynamic Mobile Manipulation Using a Large Number of Actuated Degrees of Freedom. *IEEE RAM*, 19:20–33, 2012.

Yunfei Bai Erwin Coumans. PyBullet, a Python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2020.

Anthony Francis et al. Long-range indoor navigation with PRM-RL. arXiv:1902.09458, 2019.

T. Hester, M. Quinlan, and P. Stone. RTMBA: A real-time model-based reinforcement learning architecture for robot control. In *International Conference on Robotics and Automation*, 2012.

Enrico Mingo Hoffman, Brice Clément, Chengxu Zhou, Nikos G Tsagarakis, Jean-Baptiste Mouret, and Serena Ivaldi. Whole-Body Compliant Control of iCub: first results with OpenSoT. In *IEEE/RAS ICRA Workshop on Dynamic Legged Locomotion in Realistic Terrains*, 2018.

Max Jaderberg, Volodymyr Mnih, et al. Reinforcement Learning with Unsupervised Auxiliary Tasks. In *ICLR*, 2017.

Ryan Julian, Benjamin Swanson, Gaurav S. Sukhatme, Sergey Levine, Chelsea Finn, and Karol Hausman. Efficient Adaptation for End-to-End Vision-Based Robotic Manipulation, 2020.

O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987.

S. Kim, K. Jang, S. Park, Y. Lee, S. Y. Lee, and J. Park. Whole-body Control of Non-holonomic Mobile Manipulator Based on Hierarchical Quadratic Programming and Continuous Task Transition. In *International Conference on Advanced Robotics and Mechatronics*, 2019.

Julien Kindle et al. Whole-Body Control of a Mobile Manipulator using End-to-End Reinforcement Learning. [Online] Available: arXiv:2003.02637, 2020.

Arturo Laurenzi et al. Whole-Body Stabilization for Visual-Based Box Lifting with the COMAN+ Robot. In *IEEE IRC*, pages 445–446, 2019. doi: 10.1109/IRC.2019.00092.

Sergey Levine. Reinforcement Learning and Control as Probabilistic Inference: Tutorial and Review, 2018.

Z. Li et al. Reinforcement Learning of Manipulation and Grasping Using Dynamical Movement Primitives for a Humanoidlike Mobile Manipulator. *IEEE/ASME Transactions on Mechatronics*, 23(1):121–131, 2018.

Ryan Lober, Vincent Padois, and Olivier Sigaud. Efficient Reinforcement Learning for Humanoid Whole-Body Control. In *IEEE-RAS Humanoids*, Cancun, Mexico, 2016.

M. Logothetis et al. A Model Predictive Control Approach for Vision-Based Object Grasping via Mobile Manipulator. In *IEEE/RSJ IROS*, pages 1–6, 2018.

M. V. Minniti, F. Farshidian, R. Grandia, and M. Hutter. Whole-Body MPC for a Dynamically Stable Mobile Manipulator. *IEEE Robotics and Automation Letters*, 4(4):3687–3694, 2019.

Federico L. Moro and Luis Sentis. *Whole-Body Control of Humanoid Robots*. Springer, 2018.

Sandesh Rajana. Robotics for the Supply Chain. In *2017 Third International Conference on Science Technology Engineering Management (ICONSTEM)*, 04 2018.

David Rehak, Aleš Dudáček, and Pavel Poledňák. A multipurpose robotic vehicle for the rescue of persons and interventions in emergency situations. *Komunikacie*, 15:103–109, 01 2013.

Emily-Jane Rolley-Parnell et al. Bi-Manual Articulated Robot Teleoperation using an External RGB-D Range Sensor. In *ICARCV*, pages 298–304, 2018.

Claude Sammut. When do robots have to think? *Advances in Cognitive Systems*, 1:73–81, 2012.

Cong Wang, Qifeng Zhang, Qiyan Tian, Shuo Li, Xiaohui Wang, David Lane, Yvan Petillot, and Sen Wang. Learning mobile manipulation through deep reinforcement learning. *Sensors*, 2020.

Y. Wu, P. Balatti, M. Lorenzini, F. Zhao, W. Kim, and A. Ajoudani. A Teleoperation Interface for Loco-Manipulation Control of Mobile Collaborative Robotic Assistant. *IEEE Robotics and Automation Letters*, 4(4):3593–3600, 2019.