2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Learning Arm-Assisted Fall Damage Reduction and Recovery for Legged Mobile Manipulators

Yuntao Ma, Farbod Farshidian, Marco Hutter

***Abstract*— Adaptive falling and recovery skills greatly extend the applicability of robot deployments. In the case of legged mobile manipulators, the robot arm could adaptively stop the fall and assist the recovery. Prior works on falling and recovery strategies for legged mobile manipulators usually rely on assumptions such as inelastic collisions and falling in defined directions to enable real-time computation. This paper presents a learning-based approach to reducing fall damage and recovery. An asymmetric actor-critic training structure is used to train a time-invariant policy with time-varying reward functions. In simulated experiments, the policy recovers from 98.9% of initial falling configurations. It reduces base contact impulse, peak joint internal forces, and base acceleration during the fall compared to the baseline methods. The trained control policy is deployed and extensively tested on the ALMA robot hardware. A video summarizing the proposed method and the hardware tests is available at https://youtu.be/avwg2HqGi8s.**

I. Introduction

Legged mobile manipulators have a high potential for practical applications because of their hardware capability for manipulation and traversing non-flat terrains. For such applications, specialized payloads such as sensors and end-effectors may be mounted on the robot. One factor limiting such robots' deployment is that, in case of falling, the payload and the manipulator are easily damaged, making these robots too fragile for field deployment. To this end, damage reduction during the fall and recovery from failure are considered among the key remaining challenges in the legged robotic field [1]. Both falling and recovery are contact-rich maneuvers that require the robot to make meaningful interactions with the ground. The frequent contact switches pose a challenge for the control methods, but there has recently been tremendous progress in these tasks.

To reduce the fall damage, robot controllers may plan fixed [2] or adaptive [3] contact sequences to reduce the peak acceleration, contact forces on the robot bodies, or other damage criteria. The robot may execute ukemi-like motions [4] after making contact with the ground to soften the fall. However, these methods often rely on restrictive assumptions such as inelastic and non-sliding collisions or simplifications such as falling in the sagittal or frontal plane. Furthermore, the limbs are sometimes assumed to be sufficiently agile to track the adaptive contact sequence plans [5], while this assumption does not hold for robots with heavier legs and more restrictive joint velocity limits.

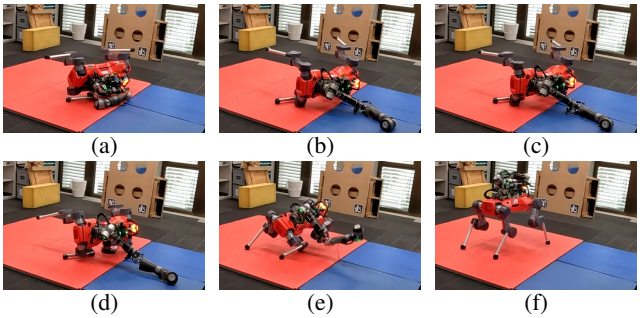


Fig. 1: ALMA robot recovering from a fall.

For fall recovery, planning-based methods often rely on accurate estimation of state and contact points between the robot and the environment, which poses additional challenges due to uncertainties in contact estimation and variations in the initial contact configuration. A recent and promising solution for the legged robot's fall recovery is to use reinforcement learning (RL) [6]–[8], where many of the heuristics and simplifications are avoided.

This paper explores a generic learning-based approach to reducing fall damage and recovering falls for legged manipulators with minimal simplifications. We extend our previous learning-based recovery methods [6], [7] by using the arm to reduce fall damage and assist in fall recovery. To this end, we train a single failing-recovery policy for legged mobile manipulators with an on-policy asymmetric actor-critic method with time-varying rewards and avoid brittle and laborious tuning of the relative weight between the motion smoothness and the time to complete the recovery. The main contributions are summarized as follows:

- A quantitative comparison between the policy and currently deployed emergency controllers (freezing and damping the drives) that shows reductions in peak instantaneous impulse on the base, peak joint internal force, and base acceleration during the fall. Ablation studies also show an improvement in recovery behavior compared to the symmetric and time-variant version of the policy.
- Simulated results that show the policy being capable of adaptively stopping the falling with the arm and performing arm-assisted recovery to resume the stance configuration within a specified time horizon. We also show hardware validations of the presented adaptive behaviors on the ALMA [9] hardware highlighted in Fig.1 in both falling and recovery scenarios.

This work was supported by the Max Planck ETH Center for Learning Systems, the Swiss National Science Foundation (SNSF) through project 166232, 188596, the National Centre of Competence in Research Robotics (NCCR Robotics), and the European Union's Horizon 2020 (grant agreement No.852044 and No.101016970). Moreover, this work has been conducted as part of ANYmal Research, a community to advance legged robotics.

All authors are with the Robotic Systems Lab, ETH Zürich, Switzerland. Email: mayun@ethz.ch

- Adaptation of our proposed method to other tasks, namely bringing the robot to the resting configuration and simple self-righting from the fall. Hardware validations of these control policies are also included.

II. Related Work

A. Fall damage reduction

Safe-falling is intensively studied in the context of humanoid robots. It is usually formulated as an optimization problem to minimize momentum change [3], [10], [11], impact force [2], acceleration [5] or a combination of damage criteria [12], [13].

Traditionally, a defined contact sequence is specified to reduce the planning variables. Ogata et al. [4] proposed implementing ukemi motion when falling forward, and Yun et al. [2] proposed tripod fall to reach early contact with the ground to reduce transferring of potential energy to kinetic energy. For the specified contact sequence, the contact location could be computed directly [4] or selected with an RL policy [2].

More recently, methods that allow for adaptive contact sequences were proposed. Ha and Liu [3] defined a Markov Decision Process (MDP) with a pre-defined contact sequence and trained a policy to minimize the vertical instantaneous impulse with the presented contact graphs. Kumar et al. [5] used RL with a mixture-of-expert structure, where each expert corresponds to a potential contact body and selects the next contact body in the contact sequence during the fall, based on each expert's value estimation.

These methods for planning multiple contacts restrict the falling to the sagittal plane to reduce the computation cost. This simplification was relaxed in [10] and [13], where the latter solves a multi-objective Quadratic Programming (QP) for post-impact to make the controller act as active compliance to reduce the damage. However, there are still other restrictive assumptions, such as inelastic contact when the robot falls.

For small humanoids, a fully-stretched arm configuration is proposed to stop falling quickly [2], [10]. However, such singularity configurations are undesirable for heavy robots such as ALMA (approximately 58 kg) because of the high stress it induces on the drives during the impacts.

B. Fall recovery

Similar to falling, recovery controllers impose many simplifications and heuristics to reduce the computational cost. Moreover, legged robots may have unreliable state estimation and uncertain contacts after falling, which pose additional challenges to optimization-based control methods. So far, most optimization-based fall recovery of legged robots usually require heuristics such as pre-defined control sequences [14], specified state transitions [15], or model simplifications [16]. These heuristics are typically designed for specific robots and require extra engineering effort to transfer onto other platforms.

On the other hand, learning-based control methods are promising alternatives because they rely less on heuristics and simplifications. Hwangbo et al. [6] and Lee et al. [7] were among the first to explore training RL-based control policies for the fall recovery of quadrupedal robots. They formulated the self-righting problem as an infinite MDP, rewarding the agent for restoring to the default resting joint position and upright orientation. A behavior selector was subsequently trained from a Finite State Machine (FSM) to hierarchically combine self-righting, standing, and locomotion for robust control of the ANYmal robot. More recently, Zhang et al. [8] proposed the K-Access algorithm for state-space clustering to improve the sample efficiency of the learning process. One remaining challenge for the RL-based fall recovery controller to this day is the arduous reward-tuning to find the right balance between the behavioral rewards and the task rewards to produce recovery behavior while avoiding jerky motions. In this work, we tackle this issue by using time-varying reward functions to reduce motion constraints before the desired recovery time.

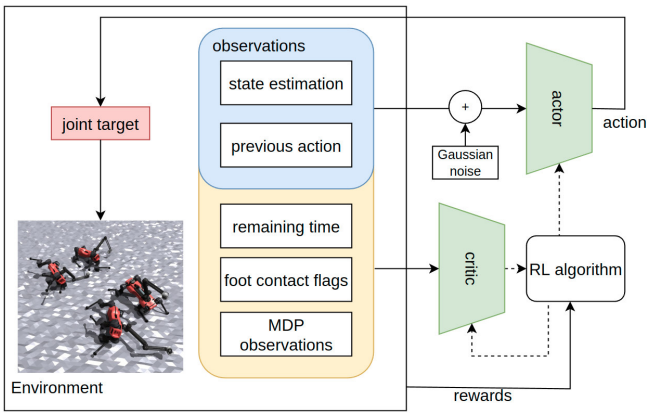


Fig. 2: Overview of the training pipeline. The actor observes minimal observation required for completing the fall recovery, and the critic has access to privileged information that improves the value function estimation. The actor's output is converted to the joint position targets for the robot's joint PD controller.

III. Method

Fig. 2 summarises our training pipeline. The control policy is trained with RL to reduce the fall damage from randomized initial falling configurations and recover within a fixed time. We formulate this task as a finite-horizon MDP where the robot receives a reward at the end of each episode based on minimizing damage criteria and recovering to the stance posture. This work employs an asymmetric actor-critic [17] training setup, where the critic receives noiseless and privileged observations. In the following, we outline the main components of our learning pipeline.

A. State initialization and rollout

Legged mobile manipulators may have different controllers for locomotion, object manipulation, or dynamic throwing tasks. As the state distribution (i.e., leg contacts, base orientation, etc.) of the robot may largely vary when

performing different tasks, it is hard to train a fall detector that suits all controllers. Therefore, we expect the task controllers to perform safety checks and report falls. Consequently, the control methods for falling should expect challenging initial conditions where parts of the body are already tripping on the ground and falling is inevitable (Fig.3a-3b). To simulate the different initial conditions of the robot falling, we randomize the initial base and joint states and disable the joint actuators for a random period in each episode ranging from 0.04 s to 1.50 s. The upper time limit, 1.50 s was chosen to allow late fall detection (robot falls completely, Fig.3c). The control policy thus learns to react in such difficult situations and could be deployed on the robot as an emergency controller that the robot automatically switches to when the task controllers detect safety violations. After the initialization period, the policy controls the robot as elaborated in III-C. The episodes only terminate at the end of the MDP's time horizon.

B. Asymmetric actor-critic

We apply asymmetric actor-critic [17] with Proximal Policy Optimization (PPO) [18] such that the critic observes privileged observation features only available in the simulation. We define the observation vectors for the actor and the critic as the following:

Actor observation: The actor observes the robot states, including the base orientation, base angular velocity, and joint states. The base linear velocity is excluded from the observation similarly to [7] because of the high uncertainty in the state estimation after the fall. Gaussian noise is added to the actor's observations during the training.

Critic observation: To improve the value estimation, the critic observes noiseless actor observations and additional privileged observations that are unavailable during the deployment but indicative of the expected return. The privileged observations include the remaining time in the episode, the foot contact states, and other observations about the MDP, including a binary flag to indicate whether the actuators are activated and the time remaining in the initial phase of rollout until the actuators become active.

In particular, note that we only observe the remaining time to the end of the episode in the critic, and the actor's policy remains time-invariant. Maximizing the PPO objective function optimizes the original finite-horizon MDP. The privileged critic produces the value function estimate for this MDP, and the limited actor observation only constrains the structure of the policy. In this problem formulation, the algorithm trains towards the optimal time-invariant policy for the finite-horizon MDP.

C. Actions

The joint target produced by the policy is computed as $(s\,a+\tilde{q})$, where s is the action scaling factor, a is the policy's action output, $\tilde{q}$ is the default joint position. The computed joint position is used as the position target for the drives' PD controller as suggested in [19]. We do not use the joint difference action (where the position target is the sum of the scaled action and the current joint positions) because for ALMA, outputting a perturbation around the specified default angle (Fig. 3d) works well as a good initial policy. The default joint positions and the action scale are selected such that random actions have a chance to flip the robot when falling sideways. This helps explore the self-righting behavior in the policy's initial training iterations.

TABLE I: REWARD TERMS SUMMARY

Reward Term	Definition		Scale
stand joint position	$\begin{cases} e^{-\frac{\sum_j (q_j^* - q_j[t])^2}{\sigma_\mathrm{p} N_j}} & \text{last 2 s} \\ 0 & \text{otherwise} \end{cases}$		350
base height	$\begin{cases} e^{-\frac{\max(h^* - h_\mathrm{b}[t], 0))^2}{\sigma_\mathrm{h}}} & \text{last 2 s} \\ 0 & \text{otherwise} \end{cases}$		600
base orientation	$\begin{cases} -g_\mathrm{b} \cdot e_z & \text{last 2 s} \\ 0 & \text{otherwise} \end{cases}$		120
body collision	$\sum_{b \in \mathcal{B}} \lVert \lambda_\mathrm{b}[t] \rVert^2$		-0.2
momentum change	$\sum_{b \in \mathcal{B}} \lVert m_b a_b[t] \rVert$		$-5e^{-3}$
body yank	$\sum_{b \in \mathcal{B}} \lVert F_b[t] - F_b[t-1] \rVert^2$		$-5e^{-2}$
action rate	$\sum (a[t] - a[t-1])^2$		$-3e^{-3}$
joint velocity	$\sum_j \dot{q_j}^2$		$-5e^{-4}$
torques	$\sum_j \tau_j^2$		$-4e^{-7}$
acceleration	$\sum_j \ddot{q}_j^2$		$-1e^{-8}$

D. Reward function

In this paper, we formulate fall damage reduction as minimizing a combination of the undesired measurements, such as high contact impulse and acceleration of bodies. In addition, the agent is rewarded at the end of the episode for standing up in a configuration close to ALMA's default stance pose. We define the fall and recovery problem as a finite-horizon MDP with time-based rewards similar to Rudin et al. [20], where time-variant task rewards are used to train efficient and adaptive locomotion skills on diverse terrains. The rewards that regularize the robot's undesirable behaviors, such as joint acceleration penalty and high impact, are time-invariant and active throughout the episode. For our problem, the rewards are categorized as follows:

a) Time-variant task rewards: The task reward for recovery contains three components. These three reward terms are activated only at the last 2.0 s of each episode.

Base height: The robot is rewarded for higher torso height. The maximum reward is obtained for height $\geq$ 0.5 m.

Joint position: Deviation from ALMA's default joint position is penalized.

Base orientation: The robot is rewarded for recovering the base orientation by penalizing the roll and the pitch angles.

b) Time-invariant behavior rewards: To encourage smooth falling, behavior rewards penalizing body collision, momentum change, and body yank, are used throughout the training episodes. The quantities required for calculation of these terms, such as the body acceleration and the contact forces, are accessed directly from the simulator.

The reward terms and the scales are summarized in Tab. I, where the task rewards are highlighted in red and the damage-reduction-related behavior rewards are in blue. In the reward definitions, q_j, $\dot{q}_j$, $\ddot{q}_j$ represent the angular position, velocity and acceleration of each joint. $\sigma_{(\cdot)}$ are the sensitivity

(a) (b) (c) (d)

Fig. 3: 3a-3c Possible initial conditions for the policy training. Our pipeline expects the task controllers to detect and report the fall, and the policy is trained to reduce the damage after switching the controller. 3d The default joint configuration for the policy training.

scaling factor of the rewards, and g_b is the projected gravity vector in the robot's base frame. $\mathcal{B}$ is the set of selected robot links, including the shanks, the thighs, the base, and all the arm links. w's are the weights of the associated rewards. m_b, a_b, λ_b, and F_b are the body mass, the body acceleration, contact force, and net force for body b respectively.

Note that for the initial phase of the episode, when the robot's actuation is disabled, both the task and the behavior regularization rewards are set to zero since the policy does not affect the robot's joint actions during the period.

E. Sim-to-Real

We use NVIDIA Isaac Gym [21] to simulate the training environment. The simulator runs at 200 Hz, and we decimate the policy to run at 100 Hz. We implemented the following techniques to facilitate sim-to-real transfer.

Actuator model: An actuator model for the leg drives is used in the simulator similar to [6]. The pseudo-direct drives in the arm have better transparency than the serial elastic actuators in the legs. Therefore, we do not implement an actuator model but only randomize the friction and add torque delays in the arm drives in each training episode.

Terrain randomization: Uneven terrain is used instead of flat terrain to randomize the ground contact normal direction and encourage larger clearance when moving above the ground.

Observation noise: Gaussian noise is added to the actor observations during the training to make the policy robust against state estimation noises on the robot.

Robot randomization: The robot's base mass is randomized with $m_{\text{rand}} \sim \text{Uniform}(-5, 5)$ kg additional mass sampled in each episode. The robot bodies' friction coefficients with the ground are also randomized.

IV. Results

A. Fall damage reduction

We evaluate the policy's performance in falling by comparing the peak instantaneous impulse on the base, mean and peak base acceleration, and peak joint internal forces across all joints. This follows the evaluation criteria in [3], [5], [22]. The max peak joint internal forces are used as one of the evaluation criteria because the high stress induced on the drives during the impact is a common cause of damage for heavy robots. The evaluation is carried out with 2560 test episode rollouts with randomized initial fall configuration and base mass, and the performance is compared to two baseline emergency controllers with the originally deployed settings, namely freezing and damping the drives.

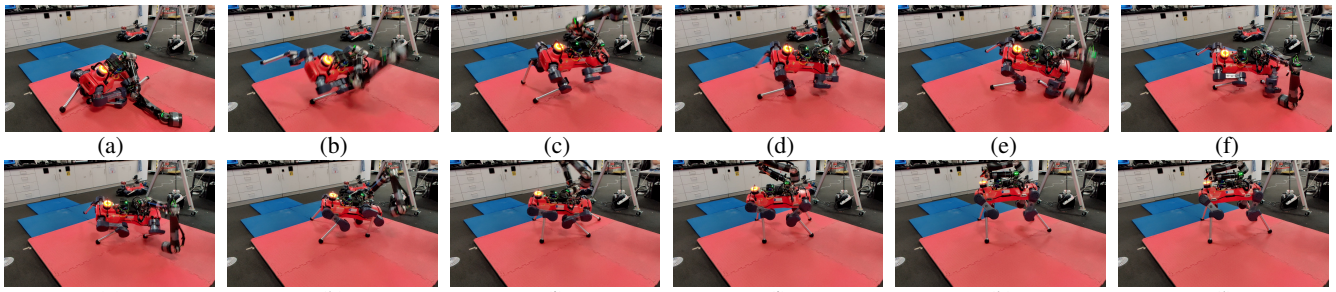
Fig. 5a shows the distribution of the base contact impulse across all time steps in all test runs. Time steps with base contact impulse below 0.05 Ns are not included in the plot because this load is not expected to damage the robot base. The lower number of samples for our proposed policy in this plot indicates that the policy avoids contact impulses above 0.05 Ns when the robot falls. In contrast, the damping controller exhibits two peaks, corresponding to the base impulse when falling directly down and sideways, respectively. Freezing the drive leads to a flat tail in the distribution, indicating a non-negligible probability of a high base impulse during the fall. The reduction in the base damage is also illustrated in Fig. 5b, where the mean and 95th percentile of base acceleration at each time step during the rollout is compared. This additionally compares the proposed method with the baseline controllers in the worst-case scenario as it is less visible in the impulse distribution plot. The 95th percentile base acceleration is significantly lower for our learning-based policy than the baseline controllers.

Fig. 6a and 6b indicate a marginal improvement in the distribution of the peak joint internal force for the proposed control policy over the damping controller and our fall-recovery controller. In both cases, we observe a significant decrease in peak internal force than the freezing strategy. Overall, our proposed method reduces the base contact impulse and the acceleration without incurring higher peak internal joint forces.

B. Fall recovery

The policy learns to use the arm adaptively for falls and recovery. Fig. 4 shows an example of the adaptive behavior when recovering from a fall. In this example, the policy first attempted a quick recovery by jumping up with the right legs and balancing by extending the left legs (4b). However, this attempt was unsuccessful, and the robot lost balance (4c). The robot then adapted its strategy and used the arm and the right-front knee to secure the fall (4d-4e). From this configuration, the robot pushed against the ground with its arm to recover (4f-4g) and subsequently retracted the arm and adjusted the stance (4h-4l). A different recovery strategy is also shown in Fig.1.

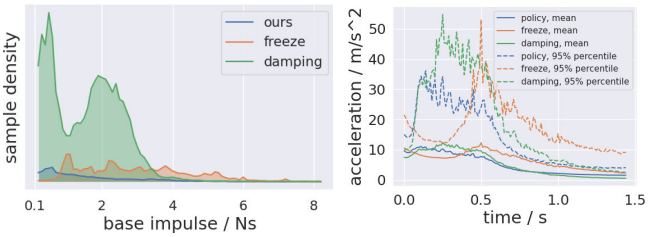
The arm-assisted recovery motion is compared to a baseline learning-based controller, where the arm is frozen and tugged throughout the episode and the policy only uses the legs for recovery. Both controllers are trained with the same reward and MDP setup except for the arm. We label a fall recovery maneuver successful if, at the end of the episode, the base height is above 0.5 m and the max joint velocity is below 0.01 rad/s. 2560 episodes were tested with each policy for comparison. We observe that the arm-assisted recovery policy achieves a 98.9% success rate compared to 95.2% with the tugged-arm policy. Fig. 7 notes the torque consumption in each drive for the falling and recovery scenarios averaged over the time steps and the test episodes. Without help from the arm, the robot needs to tug its legs further to push and



(a) (b) (c) (d) (e) (f)

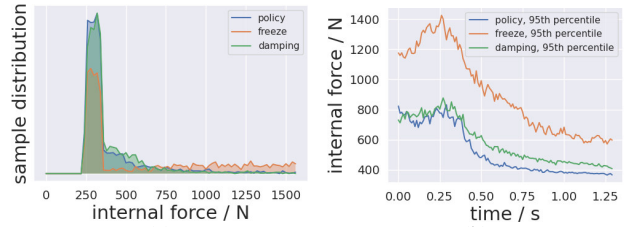
(g) (h) (i) (j) (k) (l)

Fig. 4: ALMA robot adapting the fall recovery strategy after an unsuccessful quick recovery attempt.



(a) (b)

Fig. 5: Left: Distribution of contact impulse on the base over all time steps. Time steps with base contact impulse below 0.05 Ns are not included. Right: Base acceleration during the fall.



(a) (b)

Fig. 6: Left: Distribution of peak joint internal force on the base over all time steps. Right: 95th percentile of joint internal force during the fall.

flip itself when falling sideways. We observe using the arm for recovery reduces the average leg torque consumption by 9.17% averaged over all the leg drives.

C. Ablation for the observation configurations

We compare different observation configurations for the asymmetric actor-critic setup. The configurations are summarized in Tab. II, where the evaluation values are the mean of three random seeds after training for 20000 iterations (equivalent to 3.93×10^9 environment steps). We evaluate the performance of the policies based on the episode return and the value error. The value error is the mean squared error between the value estimation from the critic and the discounted episode return. This metric indicates how accurately the critic provides the baseline function for the PPO policy update. We compare the following combinations of actor and critic observations.

a) Privileged critic: In the first configuration, the policy is trained with a critic that does not observe the episode progress or privileged robot state observations. It can be seen that this configuration is unable to learn the task. We speculate that this is because the non-privileged critic is time-invariant and thus unable to evaluate the true time-variant value function induced by time-varying reward functions, leading to significantly higher variance in the policy update. In contrast, the critic function that has access to privileged observation has a drastically lower value function estimation error, allowing the non-privileged actor to learn fall damage reduction and recovery skills.

TABLE II: COMPARING THE ACTOR AND CRITIC OBSERVATION CONFIGURATIONS

actor obs	critic obs	episode return	value error
1.(o_s)	(o_s)	-3.88	0.0902
2.(o_s) **(ours)**	(o_s, o_{priv}, o_{MDP})	12.9	0.00379
3.(o_s, o_{eplen})	(o_s, o_{priv}, o_{MDP})	12.9	0.00411
4.(o_s, o_{priv}, o_{MDP})	(o_s, o_{priv}, o_{MDP})	13.3	0.00336

o_s: state observations

o_{priv}: privileged observations (contact, noiseless state estimation)

o_{eplen}: current episode length observations

o_{MDP}: MDP observations (episode length, initialization flag)

b) Time-variant vs. time-invariant actor: Comparing configurations 2 and 3, including the remaining time of the episode in the actor's observation does not result in a significant change in the policies' episode returns, but it leads to different recovery behaviors of the actors. Fig. 8 plots the distribution of the base height over the test episodes against time and illustrates the effect of this observation on the recovery behavior. The pixels with a lighter color in the plot indicate a higher number of episodes with the corresponding height at the timestep. The time-variant policy often produces the behavior such that the robot stands halfway and rests on the joint limits of the Hip Abduction-Abduction drives. It then rapidly reorganizes its legs to the default standing configuration shortly before the task rewards are activated. We think this behavior is learned because it reduces the torque consumption in the behavior reward without lowering the task rewards. However, this is undesirable in practice because the motion is not uniform in time. Additionally, in case of a failed self-righting attempt, the time-variant policy no longer tries to recover if the expected motions penalty is higher than the expected return in the remaining time. In contrast, the time-invariant policy exhibits a temporally consistent behavior and always attempts to recover. Hence our proposed configuration is more suitable and resilient for deployment.

c) Asymmetric actor-critic vs. privileged policy: Our proposed configuration with non-privileged, time-invariant

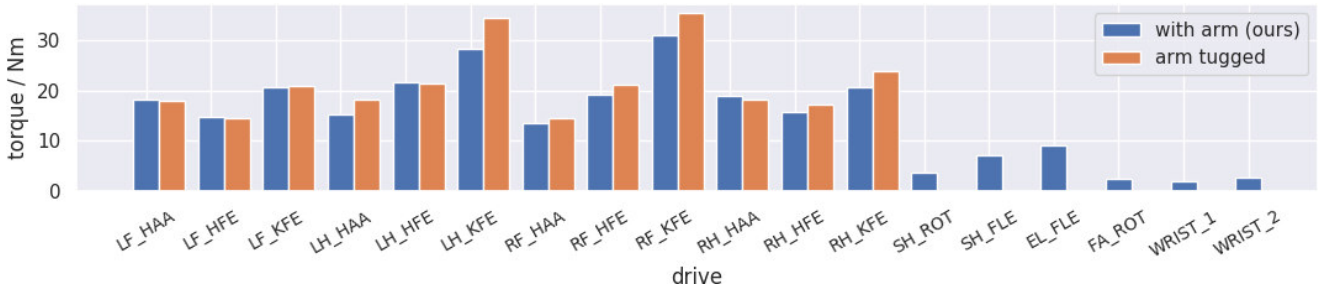


Fig. 7: The mean drive torques of the policy with and without using the arm. Both policies converged to a non-symmetric strategy of using higher torques in the Left-Hind and Right-Front legs. However, the mean torques for these two legs are lower for the policy that uses the arm.

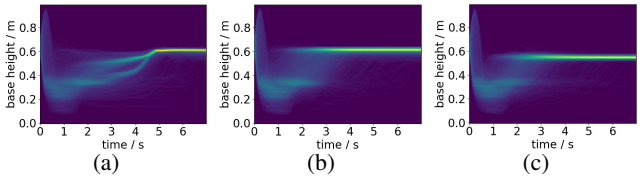


Fig. 8: The recovery profile (indicated with the distribution of base height) of the time-variant (8a) and time-invariant policy (8b) for the finite-horizon MDP. While both policies complete fall recovery within the specified time horizon, the time-invariant policy shows a more uniform behavior. Fig. 8c shows the recovery profile with the task rewards increased by three times. Its similarity to Fig. 8b indicates the robustness of the time-based reward scheme against reward scaling.

actor and privileged critic (configuration 2) results in a 3.0% decrease in the mean episode return compared to the privileged policy setup (configuration 4). However, the latter setup requires privileged actor observation and thus cannot be directly deployed on the robot. For the privileged policy, additional training steps need to be implemented to distill another policy that only uses non-privileged observation for hardware deployment as proposed in [23].

D. Reproducibility and adaptation to other tasks

We verify the robustness of the proposed method to reward scaling by comparing the recovery profiles of two policies: one trained with the original reward scale (Fig. 8b) and another trained with three times the original time-variant task rewards (Fig. 8c). Both policies learn to recover before the activation of time-variant task rewards and optimize behavior rewards, leading to similar height distributions over the episode.

We note that the training pipeline can be trivially modified for other state-transition tasks for legged mobile manipulators by only modifying high-level settings such as the rollout initialization period, the total task duration, the joint target position, and the reward scales. In the following, we outline the changes required to train policies to perform two maneuvers: resting on the ground from any stance configuration and self-righting from a fallen state to the default resting position on the floor. The two policies are also validated on hardware (shown in the linked video).

a) Resting: In the initialization period, the drives are not disabled, and the policy is rewarded for standing at a target height with randomly sampled desired joint positions. This step is required to randomize the initial stance configuration before executing the resting motion because it is hard to directly reset the robot to random stance poses on uneven terrain in the simulator. The behavior reward scales prioritize reducing the joint velocity over the torque consumption in this task.

b) Self-righting: The initialization period length is set to 2.0 s so that the robot always falls completely. The target joint configuration for this policy is the robot's default resting configuration.

For both tasks, our training pipeline discovers natural contact sequences that reduce the peak impact on the base while going to the desired final joint configuration at the end of each task.

V. Conclusion

This work presents an asymmetric actor-critic method to train a time-invariant control policy for legged mobile manipulators. The policy is trained in simulation under randomized fall configurations with time-based rewards. It learns to use the arm for falling damage reduction and recovery adaptively. For fall damage reduction, the proposed controller improves over the baselines emergency controllers in peak instantaneous impulse, base acceleration, and peak joint internal forces during the fall. It also outperforms the learning-based arm-tugged recovery policy in both recovery success rate and leg torque consumption. We extensively tested and deployed the policy in simulation and on hardware for ALMA robot.

One of the shortcomings of the current implementation is that the training environment avoids the base contacting the ground but does not adapt to potential damages such as dysfunctional drives due to the fall. We plan to extend the policy to consider these cases in the future by exploring policy architectures that allow for quick adaptation for such scenarios.

Acknowledgements

We would like to thank Joonho Lee, Nikita Rudin, Firas Abi-Farraj, and Jia-Ruei Chiu for their suggestions on the training environment. We also thank Fan Yang, Eris Sako, and Jan Preisig for their help with the experiments.

REFERENCES

[1] H. Christensen, N. Amato, H. Yanco, M. Mataric, H. Choset, A. Drobnis, K. Goldberg, J. Grizzle, G. Hager, J. Hollerbach *et al.*, "A roadmap for us robotics–from internet to robotics 2020 edition," *Foundations and Trends® in Robotics*, vol. 8, no. 4, pp. 307–424, 2021.

[2] S.-k. Yun and A. Goswami, "Tripod fall: Concept and experiments of a novel approach to humanoid robot fall damage reduction," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 2799–2805.

[3] S. Ha and C. K. Liu, "Multiple contact planning for minimizing damage of humanoid falls," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 2761–2767.

[4] K. Ogata, K. Terada, and Y. Kuniyoshi, "Falling motion control for humanoid robots while walking," in *2007 7th IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2007, pp. 306–311.

[5] V. C. Kumar, S. Ha, and C. K. Liu, "Learning a unified control policy for safe falling," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 3940–3947.

[6] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, 2019.

[7] J. Lee, J. Hwangbo, and M. Hutter, "Robust recovery controller for a quadrupedal robot using deep reinforcement learning," *CoRR*, vol. abs/1901.07517, 2019. [Online]. Available: http://arxiv.org/abs/1901.07517

[8] C. Zhang, W. Yu, and Z. Li, "Accessibility-based clustering for efficient learning of locomotion skills," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 1600–1606.

[9] C. D. Bellicoso, K. Krämer, M. Stäuble, D. Sako, F. Jenelten, M. Bjelonic, and M. Hutter, "Alma-articulated locomotion and manipulation for a torque-controllable robot," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8477–8483.

[10] A. Goswami, S.-k. Yun, U. Nagarajan, S.-H. Lee, K. Yin, and S. Kalyanakrishnan, "Direction-changing fall control of humanoid robots: theory and experiments," *Autonomous Robots*, vol. 36, no. 3, pp. 199–223, 2014.

[11] S. Wang and K. Hauser, "Real-time stabilization of a falling humanoid robot using hand contact: An optimal control approach," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE, 2017, pp. 454–460.

[12] K. Fujiwara, S. Kajita, K. Harada, K. Kaneko, M. Morisawa, F. Kanehiro, S. Nakaoka, and H. Hirukawa, "Towards an optimal falling motion for a humanoid robot," in *2006 6th IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2006, pp. 524–529.

[13] V. Samy, K. Bouyarmane, and A. Kheddar, "Qp-based adaptive-gains compliance control in humanoid falls," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 4762–4767.

[14] J. Stückler, J. Schwenk, and S. Behnke, "Getting back on two feet: Reliable standing-up routines for a humanoid robot." in *IAS*, 2006, pp. 676–685.

[15] J. A. Castano, C. Zhou, and N. Tsagarakis, "Design a fall recovery strategy for a wheel-legged quadruped robot using stability feature space," in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2019, pp. 41–46.

[16] U. Saranli, A. A. Rizzi, and D. E. Koditschek, "Model-based dynamic self-righting maneuvers for a hexapedal robot," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 903–918, 2004.

[17] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, "Asymmetric actor critic for image-based robot learning," *arXiv preprint arXiv:1710.06542*, 2017.

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

[19] X. B. Peng and M. van de Panne, "Learning locomotion skills using deeprl: Does the choice of action space matter?" in *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, ser. SCA '17. New York, NY, USA: ACM, 2017, pp. 12:1–12:13. [Online]. Available: http://doi.acm.org/10.1145/3099564.3099567

[20] N. Rudin, D. Hoeller, M. Bjelonic, and M. Hutter, "Advanced skills by learning locomotion and local navigation end-to-end," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022.

[21] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint arXiv:2108.10470*, 2021.

[22] J. Ruiz-del Solar, R. Palma-Amestoy, R. Marchant, I. Parra-Tsunekawa, and P. Zegers, "Learning to fall: Designing low damage fall sequences for humanoid soccer robots," *Robotics and Autonomous Systems*, vol. 57, no. 8, pp. 796–807, 2009.

[23] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, 2020.