

# Circus ANYmal: A Quadruped Learning Dexterous Manipulation with Its Limbs

Fan Shi\*, Timon Homberger<sup>†</sup>, Joonho Lee<sup>†</sup>, Takahiro Miki<sup>†</sup>, Moju Zhao\*, Farbod Farshidian<sup>†</sup>,

Kei Okada\*, Masayuki Inaba\*, Marco Hutter<sup>†</sup>

\* The University of Tokyo, 113-8654, Tokyo, Japan

<sup>†</sup> ETH Zürich, 8092 Zürich, Switzerland

**Abstract**—Quadrupedal robots are skillful at locomotion tasks while lacking manipulation skills, not to mention dexterous manipulation abilities. Inspired by the animal behavior and the duality between multi-legged locomotion and multi-fingered manipulation, we showcase a circus ball challenge on a quadrupedal robot, ANYmal. We employ a model-free reinforcement learning approach to train a deep policy that enables the robot to balance and manipulate a light-weight ball robustly using its limbs without any contact measurement sensor. The policy is trained in the simulation, in which we randomize many physical properties with additive noise and inject random disturbance force during manipulation, and achieves zero-shot deployment on the real robot without any adjustment. In the hardware experiments, dynamic performance is achieved with a maximum rotation speed of  $15^\circ/\text{s}$ , and robust recovery is showcased under external poking. To our best knowledge, it is the first work that demonstrates the dexterous dynamic manipulation on a real quadrupedal robot.

## I. INTRODUCTION

With the great progress on both hardware and algorithm, quadrupedal robots recently have shown significant performance in locomotion tasks, such as high-speed running [1], [2], robust falling recovery [3], [4], walking on challenging terrain [5]–[7]. Compared to wheeled or biped robots, thanks to the flexible limbs and larger support region, quadrupedal robots are more robust to cope with complex environments, such as stairs and uneven terrain. Consequently, quadrupedal robots are expected to achieve various real-world missions. These vehicles have proven to be extremely robust and can cope with complex environments and terrains, which opened the potential for application in real world missions [8], [9].

However, most quadrupedal robots’ applications focus on navigation and inspection, while active interaction and manipulation of the environment are still lacking. To overcome these limitations and extend the advanced locomotion capability with manipulation skills, several groups equipped their quadrupedal robots with a robotic arm and gripper [2], [10]–[12]. This enables basic manipulation tasks such as pick-and-place, door opening, and cooperative carrying, whereby

\* F. Shi, M. Zhao, K. Okada and M. Inaba are with JSK Lab, Department of Creative-Infomatics, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan [shifan@jsk.t.u-tokyo.ac.jp](mailto:shifan@jsk.t.u-tokyo.ac.jp)

<sup>†</sup> T. Homberger, J. Lee, T. Miki, F. Farshidian, M. Hutter are with Robotics Systems Lab, ETH Zürich, LEE building, Leonhardstrasse 21, 8092 Zurich. This research was supported by the Swiss National Science Foundation through the National Center of Competence in Research (NCCR) Robotics.

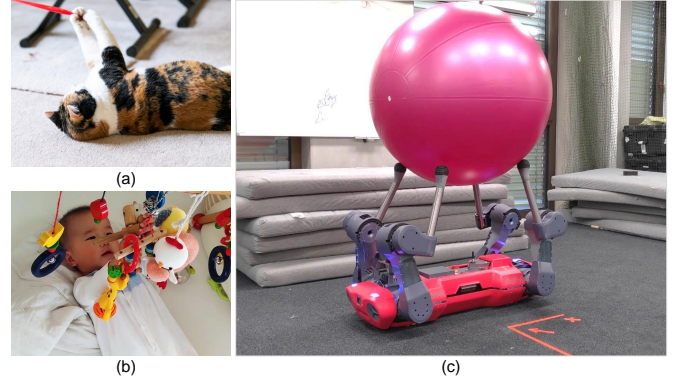


Fig. 1. Inspired by the manipulation skills of animals (a) and babies (b), we enable the quadrupedal robot ANYmal (c) to manipulate objects with its legs. Experiment video: [https://youtu.be/lmWSw\\_HI918](https://youtu.be/lmWSw_HI918)

the manipulation problem is largely decoupled from locomotion. On the downside, payload limitations of the existing quadrupedal robots mostly only allow carrying a single arm, which limits the manipulation skills.

In contrast to these developments in robotics, where locomotion and manipulation are tackled with separate hardware components, we can observe truly amazing manipulation skills in nature. Quadrupedal animals can utilize their legs to dexterously manipulate the environment (Fig. 1a). Inspired by this ability, a few approaches have suggested utilizing the legs as manipulators to achieve dexterity with minimal hardware modification. In [13], [14], two limbs are used to pick up a box and the other two limbs for stance balancing. For multi-legs robots such as hexapod robots, four limbs are used balancing while the other two limbs are manipulating objects [15]–[18]. Another solution is to add additional support legs to the quadrupedal robot to maximize the robustness of loco-manipulation [19]. Compared to dual-limbs manipulation, the most extreme idea is to synthesize all the limbs for manipulation. Inchworm-like motion with four legs is achieved in [20] to manipulate two boards. In [21], a quadrupedal robot in the simulation balances on a ball and simultaneously manipulates it.

Locomotion and manipulation are regarded as the dual problem in some aspects [22]–[24], whereby legs could be viewed as the fingers of in-hand manipulation. Dexterous in-

hand manipulation has a long history and is still a challenging problem [25]–[31]. A high-DoF hand’s controller must handle complex contact dynamics that are difficult to model accurately and challenging to compute online. Recently, deep reinforcement learning shows great progress on in-hand dexterous manipulation [32]–[39]. Robots can learn robust policies in simulation or real environment to operate valve [36], rotate block [38] and even solve the Rubik’s Cube [37] in the real world.

On the other hand, for quadrupedal locomotion, deep reinforcement learning also plays a vital role in recent progress. Walking policy is learned on a small-size quadrupedal robot [40]–[43]. Actuator network is developed to reduce the sim-to-real gap and achieve robust dynamic locomotion skills on ANYmal robot [3], [4], [7]. Gait selection is learned to cope with rough terrain with less energy usage [6]. Animal behavior is imitated and further improved by RL to deploy on the real robot [44].

#### A. Contribution

Motivated by the recent progress in learning quadrupedal locomotion and in-hand manipulation, we revisit the leg-manipulation task to attain four-limbs manipulation skills. Our main contribution is to develop a framework that enables a quadrupedal robot to achieve a first step towards dexterous full-limbs manipulation. We demonstrate our framework’s effectiveness by showcasing it on a series of challenging circus tasks such as rotating a ball in roll, pitch, and yaw direction with different velocity. To the best of our knowledge, it is the first work to achieve the quadrupedal dexterous dynamic object manipulation on a real robot.

### II. METHOD

In this paper, we propose a method for in-limb manipulation with ANYmal [45], a quadruped robot actuated with 12 series elastic actuators (SEAs), as illustrated in Fig. 2. Our method allows dynamically rotating a circus ball based on user-defined target velocities. To verify the  $SO(3)$  rotation ability, we train the policy to rotate the ball in roll, pitch, and yaw directions. Moreover, we limit the manipulation contact points only on feet, where the robot’s base should not touch the ball.

The proprioceptive measurements are limited only to the joints’ position and velocity without any contact measurement such as contact state or contact force. To further simplify the problem, position and orientation of the ball are obtained from an external motion capture system, and mass and radius of the ball are known to the controller. We train the policy with model-free RL in the Raisim simulator [46], and achieve the zero-shot deployment on the real robot.

In the rest of this section, the whole learning process would be introduced, including the environment’s Markov Decision Process(MDP) formulation, detailed training settings, network architecture, and policy gradient algorithm.

#### A. Observation and Action

The observation includes joint states, previous action, ball states, and task-related inputs. Compared to the locomotion

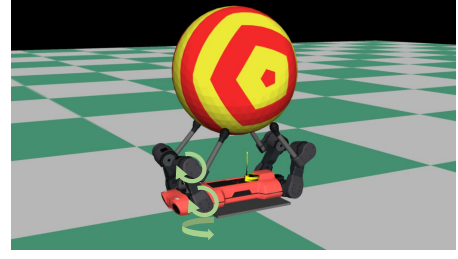


Fig. 2. Circus experiment with ANYmal-C robot in the Raisim simulator [46]. Each leg has 3 DoF driven by SEA actuators, which are Hip Abduction/Adduction (HAA), Hip Flexion/Extension (HFE), and Knee Flexion/Extension (KFE) from the bottom to the top.

task [3], [44], the position and velocity of robot base are eliminated since the robot lies on its back during the manipulation task. The joint state includes positions and velocities measured by joint encoders on the robot. The action in the previous time step (i.e., joint position command) is recorded and included as well. The ball’s position and orientation (relative to the robot base) is observed by the external motion capture system. The original command is the ball’s target angular velocity. We translate it into two parts: (1) target orientation represented by quaternion and (2) time to reach the goal orientation. Both parts are updated periodically.

We stack the observations of the last three time steps [3], [44], which is the same history length as the input to the actuator model. This enables the policy to handle latencies and partially observable states of the hardware [41]

The history of joint position  $\theta_t$ , joint velocity  $\dot{\theta}_t$ , action  $a_t$ , ball position  $p_t$  and quaternion difference to target orientation  $q_t$  are sampled at 10 and 20  $ms$  previously to avoid being too sparse or dense. Thus the observation,  $s_t \in \mathbb{R}^{130}$ , is defined as

$$\begin{aligned} x_t &= (\theta_t, \dot{\theta}_t, p_t, q_t, a_t) \\ s_t &= (x_t, x_{t-1}, x_{t-2}, t_{remain}). \end{aligned} \quad (1)$$

The action (output of the policy) is sent to the robot as the joint target position. A low-level joint position PD controller translates the policy output to motor torque command.

#### B. Reward Function

The goal of our reward design is to encourage the robot to manipulate the ball to track the commanded speed while being intuitive and simple with a minimum number of terms. In contrast to locomotion tasks [3], [44], here, the speed is implicitly represented by the periodically updated target orientation. The reasons are twofold: first, assuming sphere manipulation as the complex-terrain locomotion, it is difficult to keep constant speed [7]; and second, for most manipulation tasks, it is not necessary to maintain constant speed, but accurate tracking for target orientation is more critical [37],

[38]. Thus the reward is represented as the angle difference  $\delta q$  between the current and target orientation

$$r_t^q = k_q * \frac{1}{e^{\delta q} + 2 + e^{-\delta q}}, \quad (2)$$

where  $\delta q$  could be calculated from quaternion difference in the observation state:  $\delta q = 2 \cos^{-1}(\mathbf{q}_t(0))$ .

Aggressive leg motions can result in base motion, which is dangerous to the hardware and can cause task failure. To avoid this behavior and keep the base still, a negative reward is added to punish any base velocity.

$$r_t^v = -k_v * \|\mathbf{v}_{base}(t)\|, \quad (3)$$

here  $\mathbf{v}_{base}(t)$  denotes the linear velocity of robot base, which could be obtained in the simulation. This reward is dominant in the early training stage and reduces to zero in the middle stage.

Joint torque is punished for keeping the feasible and energy-efficient torque distribution and preventing high contact force during manipulation. High contact forces often denote that the legs are forcefully squeezing the ball, which can deform the object and leads to task failure.

$$r_t^\tau = -k_\tau * \|\boldsymbol{\tau}(t)\|^2. \quad (4)$$

Although less slippage at contact points is desirable, completely avoiding it is not realistic. It would cause a significant reality gap between simulation and real robot, which is also the limitation in the previous model-based method [21]. Thus, we only moderately punish the slipping velocity instead

$$r_t^{slip} = \begin{cases} -k_{slip} * \|\mathbf{v}_{tan}(t)\|, & \text{in contact} \\ 0, & \text{no contact} \end{cases} \quad (5)$$

where  $\mathbf{v}_{tan}(t)$  denotes the relative tangent velocity between the ball and the end-effector of each leg.

Compared to walking on the stiff ground, most of the manipulated daily object is more deformable under large contact force. Since our simulator [46] could not simulate deformation, it would lead to huge difference between the sim and real especially in the normal direction of contact forces. In our real experiment, a yoga ball is used, which is even more deformable. To tackle this problem, we punish the normal contact velocity.

$$r_t^{collide} = \begin{cases} -k_{collide} * \|\mathbf{v}_{norm}(t)\|, & \text{in contact} \\ 0, & \text{no contact} \end{cases} \quad (6)$$

where  $\mathbf{v}_{norm}(t)$  is the relative contact velocity in the normal direction between ball and the end-effector of each leg.

Compared to the reward design in quadrupedal locomotion task [4], [7], [44], [46], the joint speed reward, and foot clearance reward are eliminated. The former one is implicitly contained in the robot base and contact velocity reward. The latter one, whose essential part is the foot clearance threshold, is not intuitive to select in our setup with shaped objects compared to flat terrain.

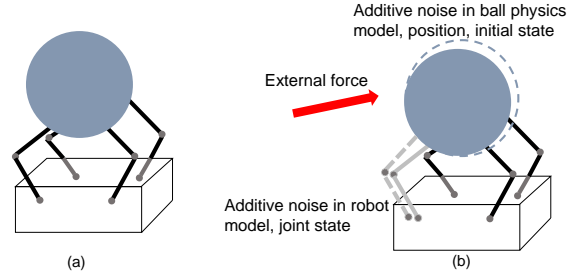


Fig. 3. Visualization of domain randomization and external disturbance force during training.

### C. Early Termination

Early termination is one of the essential components in the training procedure [47], which eliminates the local minimum or corner case with unnatural performance. In our manipulation task, the early termination would be triggered when:

- The robot gets in self-collision.
- The ball contacts with other links except for the end-effector on feet.
- The ball's position is out of a feasible region.
- The period in which the ball is in a no-contact state exceeds a threshold.

The threshold region is set as a horizontal plane being  $\pm 1.5$  radius of the ball and vertical axis being  $\pm 1$  radius. The limit on ball-no-contact time is to avoid solutions in which the robot throws the rotating ball into the air.

### D. Domain Randomization

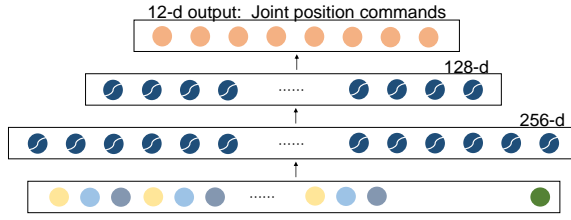
Domain randomization is a simple technique to improve a policy's robustness against modeling errors and sensor measurement noise as Fig. 3. In our circus task, the randomization takes place in the followings:

- Leg configuration. The shank positions and lengths are perturbed with additive noise  $\sim \mathcal{N}(0.03, 0)$  m.
- Joint state observation. Additive noise  $\sim \mathcal{N}(0.05, 0)$  rad to joint positions and  $\sim \mathcal{N}(0.3, 0)$  rad/s to velocities.
- Ball physical model, including mass with a random  $\nu(\pm 5)\%$  of its original weight and radius with  $\nu(\pm 10)\%$
- Ball contact model. The friction and restitution coefficients are sampled from uniform distributions  $U(0.5, 1.1)$  and  $U(0.9, 1.0)$
- Ball state observation. Additive noise  $\sim \mathcal{N}(0.04, 0)$  m to position and  $\sim \mathcal{N}(0.03, 0)$  rad to orientation in each axis
- Initialization state, including initial robot position and configuration, ball position and orientation

where the policy is encouraged to learn strategies under these varying dynamics and initialization to better deal with real world conditions.

### E. External Disturbance

Injecting random disturbance force has shown to be effective in achieving sim-to-real transfer [7], [48]. During training, the ball is applied with 50 N external force from a random



130-d input: History of joint state, ball state, target orientation, action. Remaining time

Fig. 4. Network architecture used for policy is a two-layers Multi-Layer Perceptron (MLP) with tanh activation function. The output is directly send to robot as joints position commands. The value network is similar architecture but with 1d output value.

direction. The disturbance force lasts for 0.4 s and appears at the random time point with 20 % probability.

### F. Policy Training

The policy and value network are multi-layer perceptron (MLP) with 2 hidden layers with 256 and 128 units for each and tanh activation function as Fig. 4.

The Proximal Policy Optimization (PPO) [49] algorithm is used for training. The hyperparameters are: discount factor  $\gamma = 0.998$ , clipping range  $\epsilon = 0.2$ , learning rate  $lr = 0.001$ . The parameterized policy  $\pi_\theta(a_t|o_t)$  is used to maximize the expected reward return:

$$\theta^* = \arg \max_{\theta} \mathbb{E}_{\pi_\theta} \left[ \sum_{t=0}^{\infty} \gamma^t (r_t^q + r_t^v + r_t^T + r_t^{slip} + r_t^{collide}) \right] \quad (7)$$

### G. Curriculum Learning

Curriculum learning [50] is an effective strategy to solve challenging tasks, in which the task difficulties gradually increase during the learning process. Such a technique has proven to be useful in locomotion tasks where the terrain's complexity is gradually increased in the course of training, resulting in a higher survival time in training rollouts [7], [51]. In this work, the curricula focus on the domain randomization ranges, target rotation speed, and discrete period for updating target orientation. The robot is initially trained with the low-speed manipulation commands and small additive noise in domain randomization. The target speed and noise would increase monotonically, while the updating period decreases. The initial target speed is  $0^\circ/\text{s}$  and the final speed is  $15^\circ/\text{s}$ . The updating period is set as discrete value from 1, 0.5 to 0.33 s, in which the target orientation is updated at 1, 2 and 3 Hz. To maximize total reward, the policy would generate the higher-frequency gait when the updating period decreases. The modeling and additive noise in Sec. II-D is discounted with a curriculum factor whose value would increase to 1 in the end.

## III. EXPERIMENT

The policy is trained in simulation and achieves zero-shot transfer to the real robot. We demonstrate our framework on

ANYmal-C<sup>1</sup> with a total mass of about 40 kg and 12 SEAs with a maximum torque of 80 Nm. The ball in the experiment is a commercial yoga ball with 3 kg mass and 0.8 m diameter. A VICON system<sup>2</sup> measures Center-of-gravity (CoG) position and orientation of the ball.

We have identified the following mismatches between simulation and the real robot experiments: the joints position tracking errors, the softness and deformation in yoga ball, the unexpected slip between ball surface and feet, the non-standard ellipse shape after inflation and CoG position error. However, our learned control policy has proven to be robust to these uncertainties.

To avoid the robot's feet colliding with the motion capture markers, we separately demonstrate the rotation in roll, pitch, and yaw direction with a maximum rotational speed of  $15^\circ/\text{s}$ . In the real robot, we conducted manipulation experiments for over 2 mins, during which the controller could recover robustly under external disturbance such as poking the ball.

### A. Simulation

The policy is trained in the Raisim simulator [46], which is used to simulate the rigid-body contact dynamics. During training, an actuator network [3] is used to reduce the modeling mismatch between simulation and real robots due to unmodeled actuator dynamics.

The quality of the trained policy is strongly related to both target speeds and updating periods of the target orientation. Larger target speed leads to more aggressive motion, while a shorter period leads to more frequent contact changes. The policy is trained with different updating periods, as described in Sec. II-G. The simulation results show that the policy with an extended period generates smoother motion with less contact switch but easier to fail when the target speed and domain randomization noises are increased. In contrast, the policy with a short period leads to more contact switch, but enables larger target speed with more robustness. Due to the joints' physical limitations in velocity and acceleration, the updating period could not be infinitely small. Analogous to the gait scheduling settings in locomotion, the final period is selected to be 0.33 s by updating the target orientation at 3 Hz.

We also notice that policy becomes more conservative with increasing external disturbance and growing domain randomization. One example is foot clearance decreases during training to quickly respond to the unexpected disturbance.

### B. Real Robot

We run the same policy in the real robot without any modification. The policy runs 100 Hz on the real robot and sends joints position command to the robot's joint position PD controllers.

In the real experiment, we evaluate the roll, pitch, and yaw rotation separately. For each direction, the target rotation speed is set as  $10^\circ/\text{s}$  and  $15^\circ/\text{s}$  to demonstrate the dynamic performance. To verify the robustness of the policy, we randomly

<sup>1</sup><https://www.anybotics.com>

<sup>2</sup><https://www.vicon.com>



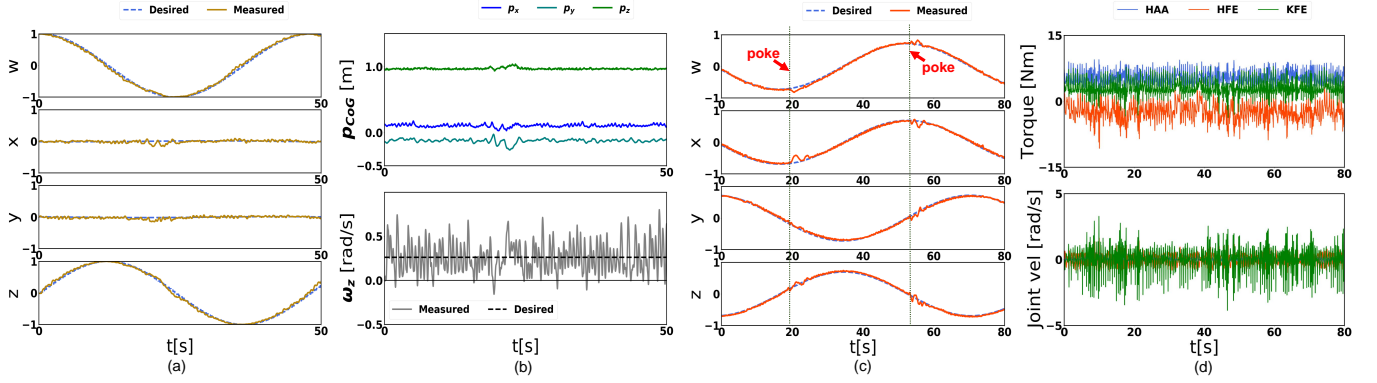


Fig. 5. Tracking results of different direction and different speeds on the real robot, in which (a), (b) show a yaw-rotation experiment with  $15^\circ/\text{s}$  speed, and (c), (d) show a pitch-rotation experiment with  $10^\circ/\text{s}$  speed under the external disturbance. Among them, (a) is the quaternion tracking results; (b) is the position and angular velocity of the ball, which are measured by the motion capture system; (c) is the quaternion tracking results under external pokes, in which the second poke is snapshoted as Fig. 7; (d) is the joint torque and velocity during the manipulation.

poke the ball and robot to simulate external disturbances during the experiments.

The tracking performance is shown in Fig. 5, in which the results of accurate tracking and recovery from the external force are presented. We randomly poke the ball and robot limbs from different directions, as the snapshots in Fig. 6 and Fig. 7. Our robot quickly recovers and continues the task, though we do not have a high-level task planner, and our non-prehensile style is less stable. Moreover, we notice in Fig. 5(b) that the angular velocity fluctuates from the target value, even if the orientation is tracked pretty well. As Fig. 5(d), the peak and averages of joints' torque and velocities during manipulation are about  $\frac{1}{4}$  of locomotion tasks using a similar ANYmal robot [3].

As for larger rotation speeds, we realize that when the target velocity is over  $20^\circ/\text{s}$ , there is more severe slippage between feet and ball because of the under-actuated legs design and limited contact area, which drives the system more likely to fail. When speeds become more enormous, the joints motion becomes more aggressive with higher acceleration, leading to a large force on the robot against the ground, which is more likely to damage the robot base.

#### IV. DISCUSSION

##### A. Locomotion and Manipulation

Multi-legged locomotion shows the duality with multi-fingered manipulation [22]–[24]. This section compares the similarity and difference between locomotion and our manipulation task under the reinforcement learning settings.

*a) Speed Reward:* In locomotion, the reward mostly directly depends on velocity [3], [4], [6], [42]. In contrast for our manipulation task, we discretize the time at a fixed frequency and update the target quaternion value based on the velocity command. Compared to direct velocity-based reward, we found that our reward design converges better during training. We speculate that the main reason is that the terrain is more complicated in our manipulation case than during flat ground locomotion, making it more difficult to track the

constant speed. A similar observation was made for rough-terrain locomotion [7]. As it is shown in the experimental results Fig. 5(b), the average velocity is pretty accurate, while the speed can be quite noisy. We also study humans conducting a similar task with a soccer ball and using their fingers. While establishing a direct parallel is impossible, we observe resembling undulant motions during the task execution.

*b) Foot Clearance:* Foot clearance denotes the distance between legs end-effector and the environment. It is a popular reward term in shaping gait behavior during locomotion to avoid foot scuffing [3], [4], [7]. The reward is often calculated by variations to a heuristic clearance value. In our manipulation task, foot clearance term is removed because it is not intuitive to decide on a heuristic value.

*c) Gait Pattern:* In a quadrupedal locomotion task, the gait pattern is often preset with specific contact sequences provided as input to the learned policy [40], [42]–[44]. In our work, we do not define any gait pattern. In contrast to legged locomotion, where the heuristics can be directly inspired by well-studied animal gaits, we cannot rely on comparable gait information for in-limb manipulation.

*d) Model-based and Model-free:* In quadrupedal locomotion tasks, model-based RL combines learned gait with a model-based whole-body controller [6]. However, due to the whole-body controller's sensitivity to model-mismatches, it was not suitable for our task as the ball deformation and slippage impede any accurate modeling.

##### B. In-Limb vs Dexterous Hand Manipulation

Our task is similar to in-hand manipulation with a dexterous robot hand. However, the fingers in the robot hand are usually driven by compact actuators with limited torque output and joint velocity. On the contrary, the actuators in our quadrupedal robot are much more powerful with respect to the manipulum's weight. The maximum torque in each SEA motor is 80 Nm, while the torque usage during manipulation is less than 5 Nm in the most time as Fig. 5(d). This difference in relative actuator power influences the reward design. While

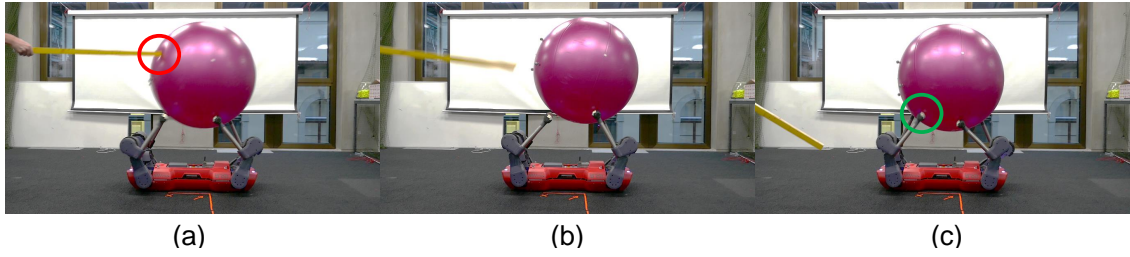


Fig. 6. Recovery from the ball being poked, while the robot is commanded to manipulate in the roll direction with  $15^\circ/\text{s}$ . (a) shows the ball being poked, in which the deformation could be noticed obviously; (b) shows the ball's state after poking; (c) shows the recovery motion generated by the policy.

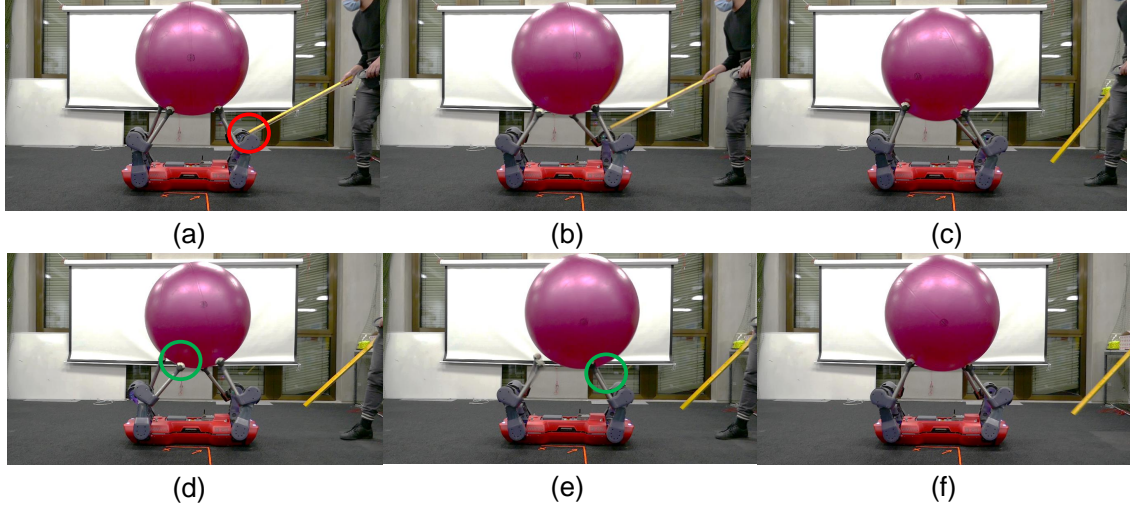


Fig. 7. Recovery from the robot leg being poked, while the robot is commanded to manipulate in the pitch direction with  $10^\circ/\text{s}$ . (a), (b) show the leg is poked with an apparent displacement in the end-effector; (c) shows influences of the contact's displacement because of the poking, in which the ball's orientation varies abnormally; (d), (e) show the recovery motion generated by the policy; (f) shows the final state after the recovery.

previous RL-based in-hand dexterous manipulation researches could design the reward with only target orientation term [36]–[38], our setup requires accounting for limbs actuation. We have realised that without penalizing the control effort, the trained policy would generate an overly aggressive motion with large joint velocity and acceleration without punishment in the reward design.

## V. CONCLUSION

This paper has proposed a novel and robust approach based on deep reinforcement learning for the quadrupedal robot to achieve a first step towards dexterous full-limbs manipulation. The policy is trained in the simulation with model-free RL, and achieve zero-shot deployment on the real robot. This is, to our best knowledge, the first work to achieve dexterous dynamic manipulation on the real quadrupedal robot.

The proposed controller exhibits dynamic manipulation performance and achieves a maximum  $15\text{deg}/\text{s}$  ball rotation speed on hardware experiments. The controller's robustness is verified on hardware by showing first, a continuous manipulation for more than 2 minutes, and second, a robust recovery under external disturbances during manipulation.

In the end, we revisit the classical argument of the duality between manipulation and locomotion. By comparing the similarities and differences in reward design under RL settings, for the challenging terrain, we hope to inspire a more closed connection on both sides.

## ACKNOWLEDGMENTS

This research was supported by the Swiss National Science Foundation through the National Center of Competence in Research (NCCR) Robotics. We appreciate the insightful discussion with Vassilios Tsounis, Bowen Yang, Jan Carius, Lorenz Wellhausen from ETH Zürich, Prof. Davide Scaramuzza from the University of Zürich, Jemin Hwangbo from KAIST, Hironori Yoshida from the University of Tokyo, Yifan Hou from CMU, Prof. Weiwei Wan from Osaka University.

## REFERENCES

- [1] B. Katz, J. Di Carlo, and S. Kim, "Mini cheetah: A platform for pushing the limits of dynamic quadruped control," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 6295–6301.
- [2] "Boston dynamics spot," <https://www.bostondynamics.com/spot>, accessed: 2020-10-10.

- [3] 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.
- [4] J. Lee, J. Hwangbo, and M. Hutter, "Robust recovery controller for a quadrupedal robot using deep reinforcement learning," *arXiv preprint arXiv:1901.07517*, 2019.
- [5] V. Tsounis, M. Alge, J. Lee, F. Farshidian, and M. Hutter, "Deepgait: Planning and control of quadrupedal gaits using deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3699–3706, 2020.
- [6] X. Da, Z. Xie, D. Hoeller, B. Boots, A. Anandkumar, Y. Zhu, B. Babich, and A. Garg, "Learning a contact-adaptive controller for robust, efficient legged locomotion," *arXiv preprint arXiv:2009.10019*, 2020.
- [7] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science Robotics*, vol. 5, no. 47, 2020. [Online]. Available: <https://robotics.sciencemag.org/content/5/47/eabc5986>
- [8] C. D. Bellicoso, M. Bjelonic, L. Wellhausen, K. Holtmann, F. Günther, M. Tranzatto, P. Fankhauser, and M. Hutter, "Advances in real-world applications for legged robots," *Journal of Field Robotics*, vol. 35, no. 8, pp. 1311–1326, dec 2018. [Online]. Available: <http://doi.wiley.com/10.1002/rob.21839>
- [9] A. Bouman, M. F. Ginting, N. Alatur, M. Palieri, D. D. Fan, T. Touma, T. Pailavanian, S.-K. Kim, K. Otsu, J. Burdick *et al.*, "Autonomous spot: Long-range autonomous exploration of extreme environments with legged locomotion," *arXiv preprint arXiv:2010.09259*, 2020.
- [10] B. U. Rehman, M. Focchi, J. Lee, H. Dallali, D. G. Caldwell, and C. Semini, "Towards a multi-legged mobile manipulator," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 3618–3624.
- [11] C. D. Bellicoso, K. Krämer, M. Stäubli, 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.
- [12] H. Ferrolho, W. Merkt, V. Ivan, W. Wolfslag, and S. Vijayakumar, "Optimizing dynamic trajectories for robustness to disturbances using polytopic projections," *arXiv preprint arXiv:2003.00609*, 2020.
- [13] T. Omata, K. Tsukagoshi, and O. Mori, "Whole quadruped manipulation," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 2. IEEE, 2002, pp. 2028–2033.
- [14] J. Hooks, M. S. Ahn, J. Yu, X. Zhang, T. Zhu, H. Chae, and D. Hong, "Alphred: A multi-modal operations quadruped robot for package delivery applications," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5409–5416, 2020.
- [15] N. Koyachi, T. Arai, H. Adachi, K.-i. Asami, and Y. Itoh, "Hexapod with integrated limb mechanism of leg and arm," in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 2. IEEE, 1995, pp. 1952–1957.
- [16] Y. Takahashi, T. Arai, Y. Mae, K. Inoue, and N. Koyachi, "Development of multi-limb robot with omnidirectional manipulability and mobility," in *Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)(Cat. No. 00CH37113)*, vol. 3. IEEE, 2000, pp. 2012–2017.
- [17] N. Koyachi, H. Adachi, M. Izumi, and T. Hirose, "Control of walk and manipulation by a hexapod with integrated limb mechanism: Melmantis-1," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 4. IEEE, 2002, pp. 3553–3558.
- [18] K. Inoue, K. Ooe, and S. Lee, "Pushing methods for working six-legged robots capable of locomotion and manipulation in three modes," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 4742–4748.
- [19] W. Wolfslag, C. McGreavy, G. Xin, C. Tiseo, S. Vijayakumar, and Z. Li, "Optimisation of body-ground contact for augmenting whole-body locomotion of quadruped robots," *arXiv preprint arXiv:2002.10552*, 2020.
- [20] G. Zhang, S. Ma, Y. Shen, and Y. Li, "A motion planning approach for nonprehensile manipulation and locomotion tasks of a legged robot," *IEEE Transactions on Robotics*, vol. 36, no. 3, pp. 855–874, 2020.
- [21] C. Yang, B. Zhang, J. Zeng, A. Agrawal, and K. Sreenath, "Dynamic legged manipulation of a ball through multi-contact optimization," *arXiv preprint arXiv:2008.00191*, 2020.
- [22] M. H. Raibert, *Legged robots that balance*. MIT press, 1986.
- [23] K. M. Lynch, *Nonprehensile robotic manipulation: Controllability and planning*. Citeseer, 1996.
- [24] A. M. Johnson and D. E. Koditschek, "Legged self-manipulation," *IEEE Access*, vol. 1, pp. 310–334, 2013.
- [25] Y. Aiyama, M. Inaba, and H. Inoue, "Pivoting: A new method of graspless manipulation of object by robot fingers," in *Proceedings of 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'93)*, vol. 1. IEEE, 1993, pp. 136–143.
- [26] A. M. Okamura, N. Smaby, and M. R. Cutkosky, "An overview of dexterous manipulation," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 1. IEEE, 2000, pp. 255–262.
- [27] A. Bicchi and V. Kumar, "Robotic grasping and contact: A review," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 1. IEEE, 2000, pp. 348–353.
- [28] I. Mordatch, Z. Popović, and E. Todorov, "Contact-invariant optimization for hand manipulation," in *Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation*, 2012, pp. 137–144.
- [29] I. M. Bullock, R. R. Ma, and A. M. Dollar, "A hand-centric classification of human and robot dexterous manipulation," *IEEE transactions on Haptics*, vol. 6, no. 2, pp. 129–144, 2012.
- [30] Y. Bai and C. K. Liu, "Dexterous manipulation using both palm and fingers," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 1560–1565.
- [31] V. Kumar, E. Todorov, and S. Levine, "Optimal control with learned local models: Application to dexterous manipulation," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 378–383.
- [32] H. Van Hoof, T. Hermans, G. Neumann, and J. Peters, "Learning robot in-hand manipulation with tactile features," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2015, pp. 121–127.
- [33] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," *arXiv preprint arXiv:1709.10087*, 2017.
- [34] G. Barth-Maron, M. W. Hoffman, D. Budden, W. Dabney, D. Horgan, D. Tb, A. Muldal, N. Heess, and T. Lillicrap, "Distributed distributional deterministic policy gradients," *arXiv preprint arXiv:1804.08617*, 2018.
- [35] M. Plappert, M. Andrychowicz, A. Ray, B. McGrew, B. Baker, G. Powell, J. Schneider, J. Tobin, M. Chociej, P. Welinder *et al.*, "Multi-goal reinforcement learning: Challenging robotics environments and request for research," *arXiv preprint arXiv:1802.09464*, 2018.
- [36] H. Zhu, A. Gupta, A. Rajeswaran, S. Levine, and V. Kumar, "Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 3651–3657.
- [37] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas *et al.*, "Solving rubik's cube with a robot hand," *arXiv preprint arXiv:1910.07113*, 2019.
- [38] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray *et al.*, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020.
- [39] A. Nagabandi, K. Konolige, S. Levine, and V. Kumar, "Deep dynamics models for learning dexterous manipulation," in *Conference on Robot Learning*, 2020, pp. 1101–1112.
- [40] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," *arXiv preprint arXiv:1804.10332*, 2018.
- [41] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, and S. Levine, "Learning to walk via deep reinforcement learning," *arXiv preprint arXiv:1812.11103*, 2018.
- [42] Y. Yang, K. Caluwaerts, A. Iscen, T. Zhang, J. Tan, and V. Sindhwani, "Data efficient reinforcement learning for legged robots," in *Conference on Robot Learning*. PMLR, 2020, pp. 1–10.
- [43] S. Ha, P. Xu, Z. Tan, S. Levine, and J. Tan, "Learning to walk in the real world with minimal human effort," *arXiv preprint arXiv:2002.08550*, 2020.
- [44] X. B. Peng, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," *arXiv preprint arXiv:2004.00784*, 2020.

- [45] M. Hutter, C. Gehring, A. Lauber, F. Gunther, C. D. Bellicoso, V. Tsounis, P. Fankhauser, R. Diethelm, S. Bachmann, M. Blösch *et al.*, “Anymal-toward legged robots for harsh environments,” *Advanced Robotics*, vol. 31, no. 17, pp. 918–931, 2017.
- [46] J. Hwangbo, J. Lee, and M. Hutter, “Per-contact iteration method for solving contact dynamics,” *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 895–902, 2018.
- [47] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, “Deepmimic: Example-guided deep reinforcement learning of physics-based character skills,” *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–14, 2018.
- [48] E. Valassakis, Z. Ding, and E. Johns, “Crossing the gap: A deep dive into zero-shot sim-to-real transfer for dynamics,” *arXiv preprint arXiv:2008.06686*, 2020.
- [49] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv preprint arXiv:1707.06347*, 2017.
- [50] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, “Curriculum learning,” in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 41–48.
- [51] Z. Xie, H. Y. Ling, N. H. Kim, and M. van de Panne, “Allsteps: Curriculum-driven learning of stepping stone skills,” *arXiv preprint arXiv:2005.04323*, 2020.