Twist-RL

Xianmai Liang, Xintong Yu, Tomas Maranga

October 2025

1 Introduction

Taking inspiration from FLEX (Forced Based Learning for Extended Manipulation, we attempt to extend its application from prismatic/revolution joints to screws. This project aims to adapt reinforcement learning onto real world robots.

Our project asks: Can we extend this idea to a third class of articulated motion, namely helical/screw motion, such as opening a bottle cap? We propose to address this challenge by learning the *force space* of the object [3]. By formulating the object dynamics into our transition function and state space, we may apply the learned force space on objects of similar dynamics.

This approach is widely applicable to a variety of robot platform. Instead of defining the joint configurations, the force and state at each time step can define the action to take for the robots.

Moreover, our approach also accounts for application towards real world robots by distributing the RL learning process into asynchronous processes [6].

2 Related Work

2.1 FLEX and one-dimensional manipulation

FLEX: A Framework for Learning Robot-Agnostic Force-based Skills Involving Sustained Contact Object Manipulation introduces a reinforcement-learning–based approach for acquiring force-based manipulation skills that generalize across different robotic platforms. [3] The framework models object dynamics with prismatic and revolute joints and leverages the built-in object dynamics provided by the Robosuite simulation environment [1] to learn a force representation for each object state. By formulating manipulation as a reinforcement learning problem in force space, the robot is able to autonomously infer joint configurations and

interaction dynamics without relying on robot-specific kinematic assumptions. This enables efficient learning and strong generalization to previously unseen objects and robot embodiments. Inspired by FLEX, our project seeks to extend this paradigm to helicoidal objects that incorporate the learning of torque. We simplified the screw problem with two-degree complexity into a one-degree problem.

2.2 Implementing RL on a Real Robots

Setting up a Reinforcement Learning Task with a Real-World Robot presents a framework for integrating reinforcement learning with physical robotic systems [6]. The authors formulate a real-world reacher task in which a robotic arm is trained to reach target positions in three-dimensional space using reinforcement learning. A central challenge addressed in the work is the synchronization between the robot hardware and the reinforcement learning loop, including state observation, action execution, and reward computation. To mitigate latency and improve stability, the system is architected with a separation between an environment thread and an agent thread. These components communicate through an actuator module that interfaces directly with the robot, continuously collecting sensorimotor data and executing a high-frequency control loop informed by the agent's actions. This decoupled design reduces communication delays between the learning algorithm and the physical robot, enabling more reliable and responsive real-world reinforcement learning.

2.3 AO grasp

AO-Grasp: Articulated Object Grasp Generation proposes a learning-based framework for generating stable and actionable grasps on articulated objects to enable downstream manipulation [7]. The method introduces an algorithm capable of reasoning over six degrees of freedom using segmented partial point clouds, allowing the model to focus on articulated object components while filtering out background geometry. This targeted representation reduces computational overhead and improves robustness in cluttered environments.

Unlike prior grasping approaches such as V-MAO, AO-Grasp does not require explicit segmentation of objects into predefined actionable parts. Instead, the model is trained via supervised learning on a large-scale dataset of approximately 78,000 articulated objects, where grasp success is defined by the detection of actionability rather than object-specific components. This formulation enables the network to generalize effectively to previously unseen objects by learning grasp-relevant features at the object level.

This work is directly relevant to our project, as AO-Grasp provides a principled mechanism for identifying stable grasp points prior to executing twisting motions. We plan to leverage this framework to initialize grasps on articu-

lated objects before force-based manipulation, potentially retraining or adapting the model to improve performance on lid-like structures, which differ from the handle-centric objects emphasized in the original study.

2.4 Unfastening Screws by a Collaborative Robot

This article addresses the challenge of automated fastener removal in remanufacturing by presenting a robust control strategy for unscrewing screws under uncertainty [4]. The proposed method employs a spiral search strategy that enables a robotic manipulator to locate and engage screws despite positional and orientational errors, a common issue in end-of-life products. Chamfer-guided engagement and visual feedback are used to facilitate alignment, while active compliance allows the system to adapt to frictional forces and minor misalignments that would otherwise cause rigid control strategies to fail.

The integration of torque sensing with adaptive motion control enables reliable detection of screw engagement and release, resulting in a reported success rate of 98%. Although the approach does not rely on reinforcement learning, it provides a detailed analysis of six-degree-of-freedom motion and force–torque interactions during the unscrewing process.

This work is directly relevant to our research, as it offers valuable insight into using torque feedback to infer successful screw disengagement. The authors' treatment of compliant 6-DoF motion serves as a useful foundation for formulating our own learning-based twisting and unfastening policies.

3 Technical Background

3.1 Simulation Platform - MuJoCo and Robosuite

FLEX frames sustained-contact manipulation as two MDPs $M_p = \langle S_p, A, R, T_p, \gamma \rangle$ and $M_r = \langle S_r, A, R, T_r, \gamma \rangle$, where the state encodes the joint axis as a 3D unit vector plus a small amount of task progress, the *action* is a bounded force vector, and the *transition* is provided implicitly by MuJoCo [5]. The object and force-centric formulation makes the policy robot-agnostic. In MuJoCo, prismatic and revolute joints are native, so the transition functions T_p and T_r are simply calls to the physics engine.

For helical motion, MuJoCo's documentation suggests composing a hinge joint and a slide joint and then coupling them with a joint equality of the form $q_{\text{slide}} = \text{pitch} \cdot q_{\text{hinge}}$, which will effectively yield a screw joint. Our project follows this strategy. We build a helical joint in simulation and train a policy on top of that fixed simulator transition.

After training the policy, we deploy the trained model onto Robosuite for simulation, using a Franka Emika Panda arm. We created a bottle model where we used its parameters to train the model on. Figure 1 is a screenshot of the finalized simulation of the bottle using Robosuite.

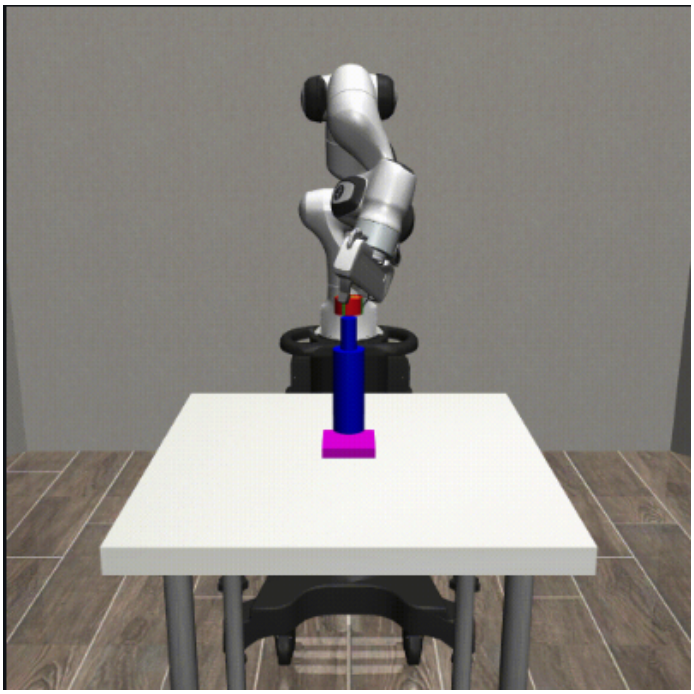

Figure 1: Screenshot of Robosuite simulation using Franka Emika Panda arm and bottle model

3.2 RL algorithm - TD3

Twin Delayed Deep Deterministic Policy Gradient (TD3) is an actor-critic reinforcement learning algorithm for continuous control. It learns two functions: an *actor* $\pi_\phi(s)$ that maps states to actions, and a *critic* $Q_\theta(s,a)$ that estimates the expected return of taking action a in state s. The critic is trained to satisfy the Bellman equation $Q(s,a) = r + \gamma Q(s',a')$, while the actor is trained to maximize the critic's estimate via the policy gradient $\nabla_\phi J = \mathbb{E}[\nabla_a Q(s,a)|_{a=\pi_\phi(s)} \nabla_\phi \pi_\phi(s)]$.

TD3 introduces three improvements over its predecessor DDPG. First, **clipped double Q-learning** uses two critics and takes their minimum $y = r + \gamma \min(Q_{\theta'_1}, Q_{\theta'_2})$ to combat overestimation bias. Second, **delayed policy updates** update the actor less frequently than the critics (every $d = 2$ steps), allowing Q-estimates

to stabilize. Third, **target policy smoothing** adds clipped noise to target actions $\tilde{a}' = \pi_{\phi'}(s') + \text{clip}(\epsilon, -c, c)$ to prevent exploiting Q-function peaks.

4 Experimental Design

4.1 Assumptions and Setup

We make several assumptions to keep the problem focused on learning the wrench policy:

- The robot's end-effector is already in contact with and stably grasping, the bottle cap/screw. We do not learn grasping or approach in this project.
- Pre-breakaway. We encoded friction at the start of the unscrewing process to simulated the initial, tightly fastened portion.
- Known screw axis. We assume we can express the screw axis in the world/object frame as a 3d unit vector $h \in \mathbb{R}^3$
- Sim first. All training is in MuJoCo/Robosuite with a helical joint created as hinge + slide with equality.
- We don't account for the items mass, therefore we don't account for linear acceleration or angular acceleration

4.2 Markov Decision Process

After studying the screw theory [2], we model the task as an MDP

$$M_h = \langle S_t, A, R, T, \gamma \rangle$$

specialized for helical motion.

State space S_t. Our base state is

$$s_t = [h, \boldsymbol{p}]$$

where:

- $h \in \mathbb{R}^3$is the unit vector for the axis of the screw. In our case, we assume that the axis is vertical, which defaults to $[0, 0, 1]$
- $\boldsymbol{p} \in \mathbb{R}^3$ is the relative position of the contact point with respect to its original position.

Action Space A**.** Instead of a 3d force, our action is the axial torque

$$a_t = \tau_z$$

We bound the action as

$$\tau_z \in [-\tau_{max}, \tau_{max}]$$

which will allow us to scale to [0,1].

Transition T**.** We do not learn T. The environment advances via MuJoCo, helical constraint turns applied wrench into coupled rotation/translation and returns next state (θ_{t+1}, z_{t+1}).

Reward Function R**.** The reward function feeds back a combination of three types of rewards.

1. **Success reward** - A large reward given after the bottle is successfully unscrewed.
2. **Reward according to progress** - Calculated with respect to the relative position to the original position. This value is calculated as a percentage of the success reward.
3. **Negative force reward** - As the goal of training is to find the smallest torque possible, we assign negative value with respect to the torque.

Discount γ**.** We value long-term rewards almost as much as short-term rewards. This is because success of both screwing and unscrewing takes long-term effort and sustained interaction. By setting our discount factor to 0.99, we value future rewards only slightly less than current reward.

5 Experimental Results

We evaluate our approach using the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm to learn continuous control policies for the twisting task. Training performance is analyzed using episode return, critic loss, and average Q-value to assess policy convergence, value estimation stability, and learning dynamics.

5.1 Learning Performance and Policy Convergence

Figure 2 shows an interesting training dynamic. The agent quickly learns a good policy early on (reaching 132 reward by episode 50), but then experiences a significant performance collapse between episodes 100-175, dropping to around 116. This dip likely indicates the agent exploring suboptimal regions of the policy space or encountering instability in learning. Encouragingly, the

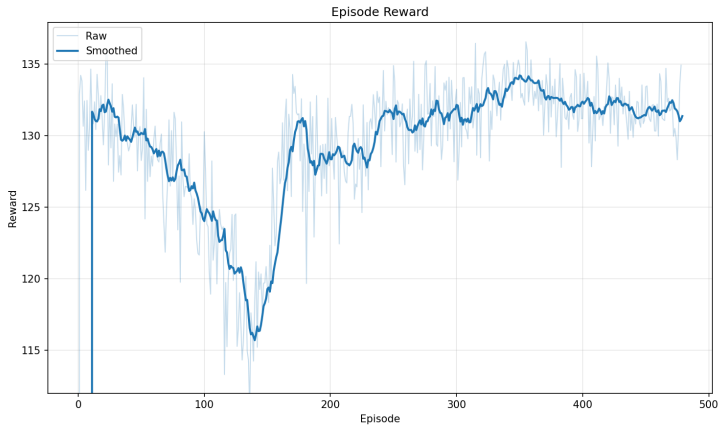

Figure 2: Episode reward over training episodes.

agent recovers and stabilizes around 131-133 reward after episode 200, suggesting eventual convergence to a near-optimal policy. The persistent variance in raw rewards throughout training indicates the environment has inherent stochasticity, though the smoothed trend confirms stable learned behavior.

5.2 Critic Loss Dynamics and Value Estimation Stability

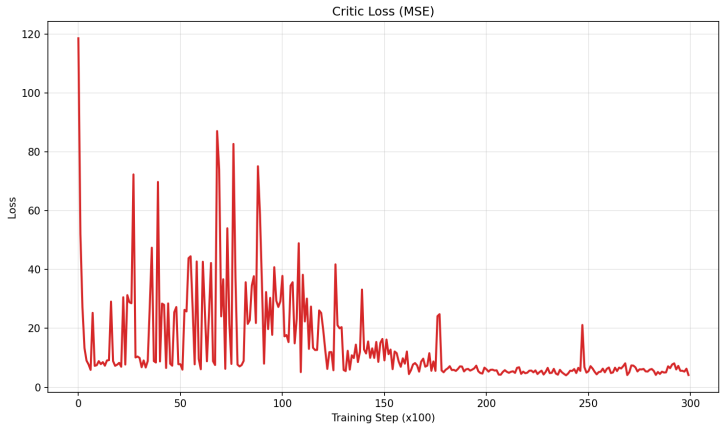

Figure 3: Critic MSE loss during training.

The evolution of the critic mean-squared error (MSE) loss is shown in Figure 3. The critic loss curve reveals the value function's learning trajectory in this actor-critic architecture. Early training (steps 0-15,000) shows high volatil-

ity with frequent spikes reaching 70-120, reflecting the critic's initial struggle to accurately estimate state values as the policy rapidly evolves. The period of intense fluctuation between steps 5,000-12,000 corresponds to the reward collapse seen in Figure 1, suggesting the critic's instability contributed to poor policy updates during that phase. After step 15,000, the loss progressively stabilizes and settles below 10 by step 20,000, indicating the critic has learned reliable value estimates. This convergence to low, stable MSE loss confirms the value function has reached a consistent approximation, enabling the actor to receive accurate gradient signals for policy improvement.

5.3 Q-Value

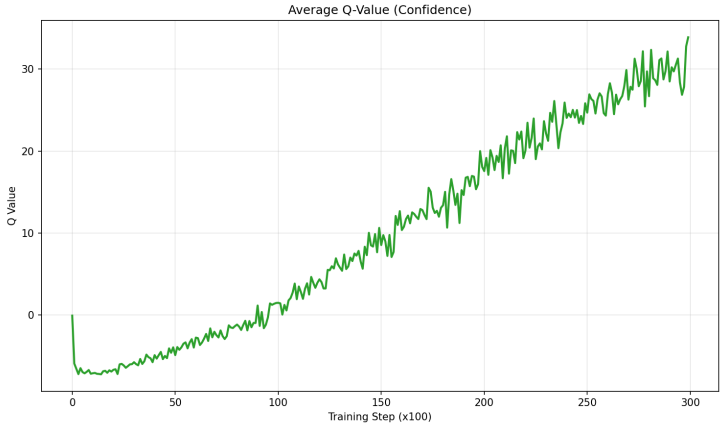

Figure 4: Average Q-value across training episodes.

In Figure 4, the average Q-value trajectory demonstrates the agent's growing confidence in its learned policy over training. Initially, Q-values hover around -5 to 0 during the first 10,000 steps, reflecting the critic's pessimistic or uncertain estimates before sufficient experience is gathered. A steady upward trend begins around step 10,000, with Q-values climbing approximately linearly through training to reach 30+ by step 30,000. This continuous increase indicates the agent increasingly expects higher cumulative future rewards from its actions, consistent with the improved and stabilized policy observed in the reward curve. However, the unbounded growth warrants attention—while rising Q-values typically signal learning progress, excessively high values without corresponding reward improvements could indicate Q-value overestimation, a known issue in actor-critic methods. In this case, the growth appears healthy given the concurrent reward stabilization seen earlier.

5.4 Discussion

Overall, the experimental results demonstrate that TD3 can rapidly learn a stable policy for the twisting task, achieving reliable performance after relatively few training episodes. However, the observed reward plateau and increasing Q-values suggest that the learned policy represents a locally optimal solution and that additional incentives are required to encourage continued performance improvement. These findings highlight the importance of reward design and exploration strategies when applying continuous-control reinforcement learning to contact-rich manipulation tasks.

6 Future Work

We trained an object agnostic policy that fits any object with a helicoidal joint. Ideally, with limited knowledge and perception of the helicoidal joint, our model can efficiently predict the unknown parameters of the joint and complete the task with a learned 6D-force space that maps directly to a specific object it's provided with.

Currently, our model works in Robosuite simulation with Franka Emika Panda arm. In the future, we plan to test it on other models in simulation and also deploy the model onto real world robots and objects. Future work would also address the twisting of similar objects that incorporates friction, object mass, grasp positions and different shapes of handles.

References

[1] Overview — robosuite 1.5 documentation, 2017.

[2] Marco Carricato. Kinematics statics freedoms screw systems iss and pss mobility design singularities screw theory and its applications in robotics, 2017.

[3] Shijie Fang, Wenchang Gao, Shivam Goel, Christopher Thierauf, Matthias Scheutz, and Jivko Sinapov. Flex: A framework for learning robot-agnostic force-based skills involving sustained contact object manipulation. pages 4782–4788, 05 2025.

[4] Ruiya Li, Duc Truong Pham, Jun Huang, Yuegang Tan, Mo Qu, Yongjing Wang, Mairi Kerin, Kaiwen Jiang, Shizhong Su, Chunqian Ji, Quan Liu, and Zude Zhou. Unfastening of hexagonal headed screws by a collaborative robot. *IEEE Transactions on Automation Science and Engineering*, page 1–14, 2020.

[5] DeepMind Technologies Limited. Overview - mujoco documentation.

[6] A Rupam Mahmood, Dmytro Korenkevych, Brent J Komer, and James Bergstra. Setting up a reinforcement learning task with a real-world robot. *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4635–4640, 10 2018.

[7] Carlota Parés Morlans, Claire Chen, Yijia Weng, Michelle Yi, Yuying Huang, Nick Heppert, Linqi Zhou, Leonidas Guibas, and Jeannette Bohg. Ao-grasp: Articulated object grasp generation. pages 13096–13103, 10 2024.