In-Hand Manipulation via Deep Reinforcement Learning for Industrial Robots

Leonardo V. O. Toledo(✉), Gustavo Jose Giardini Lahr, and Glauco A. P. Caurin

Sao Carlos School of Engineering, Av. Trabalhador São-Carlense, 400, São Carlos, São Paulo, Brazil
{toledo.leo,gustavo.lahr}@usp.br, gcaurin@sc.usp.br

Abstract. Robotics manipulation is still a challenge in many scenarios, especially when the orientation of the tool or part is determinant for task success. A study with pivoting, an in-hand manipulation technique, was conducted, which consists in re-orienting the part around one rotational axis without dropping. It gets more complex in industrial robots, which are position controlled and often the user does not have access to the dynamic parameters. Deep reinforcement learning has been successful with model free approaches as it learns the behavior of the whole system. A simulated experiment for pivoting was conducted for part alignment to a desired angle with a one degree of freedom robot and a parallel gripper. Position control and torque control were simulated and compared.

Keywords: Robotics manipulation · Pivoting · Deep reinforcement learning

1 Introduction

Many tasks require picking an object in different positions, but the initial grasp may deviate it from the desired gripping angle. This variation occurs from many factors, such as distinct initial positions from both gripper and object, changes in friction and variations on the final position profile. Solutions to this problem are re-grasping or the use of in-hand manipulation techniques.

Re-grasping restarts the task and try a new pick to minimize the gripping error [2, 7], which can be very time consuming or not even possible. Meanwhile, in-hand manipulation aims to solve the problem when the object still on the gripper [5]. Although the positive efficiency, these methods require high mechanical complexity, which is expensive and time consuming for industrial tasks.

In this paper, we study the pivoting strategy as an in-hand technique for tool-part reorientation [2]. Pivoting an object consists of rotating it along a single axis to reorient it to the desired angle. It is very useful specially with parallel grippers, which have a broad use in industry. Control alternatives for pivoting would be the implementation of underactuated control of manipulators [9], as the robot would be seeming as the actuated and the tool or part, not actuated. However, precise models of the robot or the parts are not easily obtainable in industrial setups. Industrial robots commonly have their dynamics

M. Pucheta et al. (Eds.): MuSMe 2021, MMS 94, pp. 222–228, 2021.
https://doi.org/10.1007/978-3-030-60372-4_25

not known by the user, so friction and size may vary in different scenarios. Also, it is also difficult to obtain a good friction model for part-tool and robot interaction, especially for robust simulations.

Deep reinforcement learning has presented itself as a good candidate to solve manipulation problems [3, 6]. However, these setups are expensive and sensor rich, which is not always the case for industrial applications. Other in-hand manipulations have fewer complex setups, but they usually use torque-controlled joints, and this option is not available in industrial robots [4].

We present a deep reinforcement learning setup to learn in-hand manipulation for positioning part with a parallel gripper in industrial robots. Using Proximal Policy Optimization as reinforcement learning algorithm, we compare the torque-controlled approaches with the proposed position controlled and show the similar learning behavior.

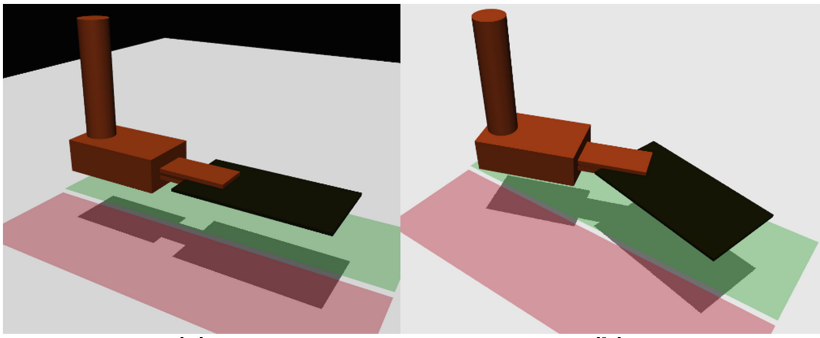

Fig. 1. Rigid bodies model built in Mujoco. Image a) shows the system before a task has started and b) after it is completed

2 Modeling and Control

This section provides a formalization of the problem and describes the modeling and control. The goal is to perform the pivoting from an initial angle to the desired angle as shown in Fig. 1. A parallel gripper holds the part, while the arm can either rotate along a single axis generating inertial forces to move the part or control the holding pressure on it by moving its fingers, varying the normal force. We assume that the task to be solved starts already with the part grasped.

Deep Reinforcement Learning algorithms are good candidates to control the task. Their negative point is the time it takes to learn in real models, since steps like resetting the robot on each episode may be very time-consuming. To deal with this setback, we propose a previous phase, using a simulation to speed up training. Further, the trained network can be transferred to real applications using transfer learning techniques [11]. We present a robust simulation model, build with Mujoco and its training using the Proximal Policy Optimization [8] algorithm.

2.1 Rigid Bodies Modeling

We used Mujoco to build the rigid body model. The main body is composed of a box B_{main} that contains a revolution joint R, limited to $\pm 40°$ and uses a proportional gain of $Kp = 30$. A two-finger gripper is used and modeled by two prismatic joints, and thus controls the pressure over the part, with a limited range of 2.5 mm and $Kp = 80$. A free joint is attached to the part, meaning it is free to assume every possible position in space. Training was conducted with a part of 215 × 120 × 4 mm and 0.82 kg. The system stands on a horizontal plane with gravity acceleration of -9.81 m/s^2. The sliding friction coefficient between gripper and part is 0.85. The torsional friction, acting around the contact normal, has a value of 0.004 between each body [4]. To enforce industrial robots' applications, we used a position-controlled actuator. For comparing purposes, we trained a similar model with the same parameters but with a torque-controlled actuator using with a gear ratio of 20.

2.2 Control Strategy

In recent years, reinforcement learning algorithms have been increasing its applications in robotics, demonstrating great potential for solving challenging decision-making problems. In this section, we provide proper background for Markov Decision Process (MDP) [10] and Proximal Policy Optimization (PPO) [8]. The goal of any reinforcement learning algorithm is to solve an MDP, defined as a 4-dimensional tuple (S, A, Pa, Ra), where S and A are the finite space of states and actions, respectively; $\boldsymbol{s}_t$ and $\boldsymbol{a}_t$ are the observed space and action taken at time t.

The core problem of an MDP is to find a policy π which defines the action $\pi(s_t)$ that our agent will choose at state s_t. To solve the MDP problem, we address the Proximal Policy Optimization (PPO), implemented using module SpinningUp [1], an on-policy algorithm that takes the biggest possible improvement step without causing a performance collapse. PPO updates policies via Eq. (1), where θ is the policy parameter, E_t denotes the empirical expectation over timesteps, A_t is the estimated advantage at time t, $\in$ is a hyperparameter (usually 0.2) and r_t is the probability ratio under the new and old policies. The term $clip(r_t(\theta), 1 - \in, 1 + \in)A_t$ in Eq. (1) works as filter of the probability ratio, removing the incentive of moving r_t outside of $[1 - \in, 1 + \in]$, therefore preventing possible policy degradation.

$$L^{CLIP}(\theta) = E_t\big[min(r_t(\theta)A_t, clip(r_t(\theta), 1 - \in, 1 + \in)A_t)\big], \tag{1}$$

3 Learning

The Observation Space is composed by the measured variables defined by (2) and the description of each state is given by Table 1.

$$s_t = \big[\theta_{part} - \theta_{tgt}, d_{dist}, \dot{\theta}_{grp}, \dot{\theta}_{part}, drop\big], \tag{2}$$

The first parameter, $\theta_{part} - \theta_{tgt}$, gives us the current distance from the goal, allowing the network to generalize different policies for different distances. The second one, d_{dist},

Table 1. Description of each element of Eq. (2)

Element	Description
θ_{part}	Part's angle with relation to the gripper
θ_{tgt}	Target angle
d_{dist}	Distance between B_{upper} and B_{lower}
$\dot{\theta}_{grp}$	Gripper's velocity
$\dot{\theta}_{part}$	Part's velocity with relation to the gripper
drop	Boolean variable indicating if the part dropped from the gripper

controls the pressure over the part, exerted by the gripper. The third and fourth, $\dot{\theta}_{grp}$ and $\dot{\theta}_{part}$, gives the correct velocity values that generate the expected dislocation of the part. The drop allows the agent to tell if it is holding the part or not.

The Action Space at each time step t is given by $\boldsymbol{a}_t = \left[\theta_{grp}, d_{dist}\right]$. The first element, θ_{grp}, represents the gripper's position. With these 2 actions our agent can accomplish the objective. Our method validates its proficiency for industrial assembly, where more complex solutions may be impracticable regardless the success rate. The reward works with 3 distinct rules, shown in (3). The base rule is present from the beginning of the simulation until the part reaches the success zone, that is, the region $\theta_{tgt} \pm \Delta\theta$, where Δ_θ is the acceptable error. Once the part is at the success zone $\eta = 2$, otherwise $\eta = 1$. This generate extra incentive of remaining inside this region. To minimize the difference from θ_{tgt} and make sure of the stability, the agent must stay in the success zone for 120 timesteps. The third and last rule applies after reaching the time restriction, when the episode ends, and the agent receives $R_t = 10$.

$$r_t = \begin{cases} -\frac{|\theta_{part} - \theta_{tgt}|}{100\eta}, & \theta_{part} \in \left(-\infty, \left(\theta_{tgt} - \Delta_\theta\right)\right) \cup \left(\left(\theta_{tgt} + \Delta_\theta\right), +\infty\right) \\ 10, & \theta_{part} \in \left[\left(\theta_{tgt} - \Delta_\theta\right), \left(\theta_{tgt} + \Delta_\theta\right)\right] \end{cases}, \tag{3}$$

The success zone is given by (4):

$$\Delta_\theta = \begin{cases} \frac{|\theta_{tgt}|}{7} + 0.3, & -6 \leq \theta_{tgt} \leq 6 \\ \min\left(\frac{|\theta_{tgt}|}{7}, 3\right), & \theta_{tgt} \langle -6 \cup \theta_{tgt} \rangle 6 \end{cases}, \tag{4}$$

Using a variable error, instead of fixed 3° for instance, forces the agent to be more precise for smaller angles and allows it to be able to adequate for larger ones. The limit of 3° holds this adequacy from going too far and undermine the simulation. In Sect. 4, Fig. 4 illustrates Eq. (4), defined above. The learning module used is SpinningUp [1]

4 Results

4.1 Training

The training occurred within 2000 episodes. Each episode has the maximum duration of 4000 timesteps. The number of timesteps per epoch was set to 8000. A multi-layer

perceptron network was used for function approximation. Various architectures were tested, but it responded best with 3 hidden layers of 32, 24 and 10 neurons, respectively. We used the discount factor of $\lambda = 0.99$. Figure 2 shows the mean reward of the position-controlled actuator in blue and torque controlled in red. The system converges to the object faster in the position-controlled model and remains more stable during the whole training. It is worth mentioning that all the conditions were kept the same for both models, changing only the used actuator.

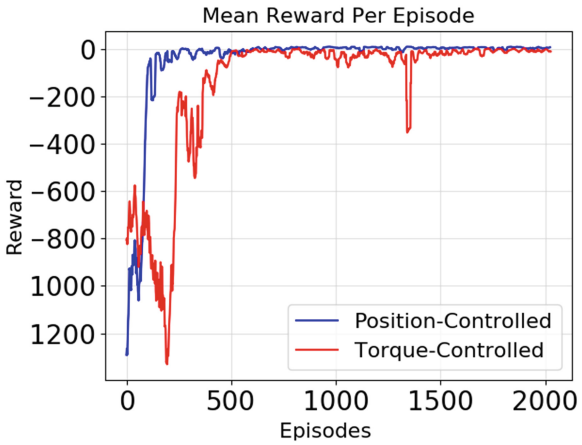

Fig. 2. Mean reward per episode during training phase

4.2 Experiments

In this section, we present the results for 2 different comparisons. The first one compares practical results of both presented actuators, and the second one reveals the robustness of our model for different parts.

Since the reward plot in (2) shows a mean value, it can hide some fails. The model may work very well for some range of angles and fail completely in other range. To analyze that, Fig. 3 shows the error in degrees when trying to reach a target angle. The green area represents the success zone, defined in Eq. (4). For each target angle, 30 runs were done, and then we extract the mean value of it. All the results so far were collected using a single part. To confirm the generalization of the model, we must verify the task execution on different parts. Therefore, two extra parts were tested: part 2 with dimensions 120x100x4 mm and 0.38 kg, and part 3 with 180x220x4 mm and 1.27 kg. The friction was kept the same as the previous experiments.

Figure 4 presents the error for all parts over the target angle. Meanwhile the trained part was capable to achieve all desired angles under the acceptable errors, due to its high fidelity to original data, the other had problems and started to deviate with more than 20° from initial position, for either parts. This happens since the manipulator needs to move faster to achieve the target angle, thus applies higher accelerations, leading to not

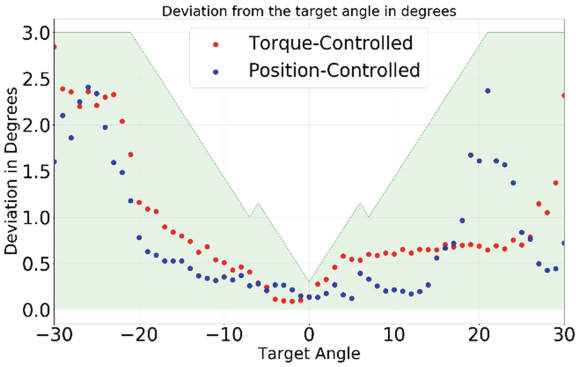

Fig. 3. Deviation from the target angle when in degrees with 2 different actuators

negligible inertial components, which is harder to generalize. This behavior is function of the inertia of the parts, but also to the friction in interaction between grippers and part, that is more complex to model and has a wide range of values. For most of angles range, different parts confirm the model flexibility, while it is possible to observe failures for larger angles. In future work, we consider adding different parts during the training process and include variables like mass and part dimensions to the observation space.

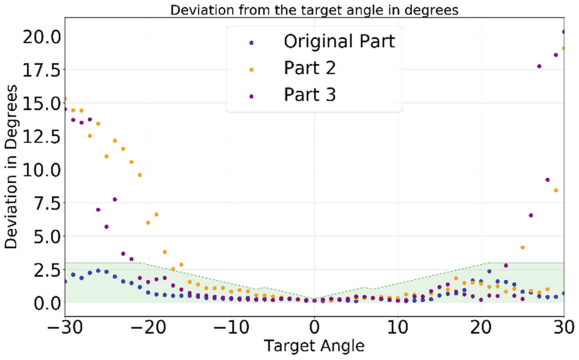

Fig. 4. Deviation from the target angle in degrees for 3 different parts

5 Conclusions

This works proposes the use of reinforcement learning algorithms to solve in-hand manipulation problems for industrial robots, focused on pivoting. Pivoting consists of the reorientation of the object by generating inertial forces to move the part. The model was implemented using a precise physics simulator, Mujoco. The learning algorithm applied to solve the task was Proximal Policy Optimization and the actuation was executed by a position controller, present in ubiquitously present in industrial automation systems,

including robot manipulators. The model achieved a high level of success for position and torque actuators, corroborating to its robustness. All tests with the training part achieved small errors. When evaluated for generalization with different parts, the trained model was able to generalize for angles closer to the initial one, i.e., range of [− 20°, 20°]. On future work, the implementation of a variety of parts during training and the inclusion of its parameters in the observation space suggest a possible improvement in the generalization. Also, we would like to evaluate the effect of friction during training and generalization of the parts, and validate the simulated model on a real robot.

Acknowledgements. The authors would like to acknowledge CNPq for the processes 314936/2018-1 and 141395/2017-6, São Paulo Research Foundation (FAPESP) grant 2017/01555-7, and University of São Paulo for USP-PRP Call 668. This study was partially funded by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) – Finance Code 001.

References

1. Achiam, J.: Spinning up in deep reinforcement learning (2018)
2. Aiyama, Y., Inaba, M., Inoue, H.: 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 1993), vol. 1, pp. 136–143. IEEE (1993). https://doi.org/10.1109/iros.1993.583091. http://ieeexplore.ieee.org/document/583091/
3. Andrychowicz, O.M., Baker, B., Chociej, M., Józefowicz, R., McGrew, B., Pachocki, J., Petron, A., Plappert, M., Powell, G., Ray, A., Schneider, J., Sidor, S., Tobin, J., Welinder, P., Weng, L., Zaremba, W.: Learning dexterous in-hand manipulation. Int. J. Robot. Res. **39**(1), 3–20 (2020). https://doi.org/10.1177/0278364919887447
4. Antonova, R., Cruciani, S., Smith, C., Kragic, D.: Reinforcement learning for pivoting task. arXiv preprint arXiv:1703.00472 (2017)
5. Chavan-Dafle, N., Mason, M.T., Staab, H., Rossano, G., Rodriguez, A.: A two-phase gripper to reorient and grasp. In: IEEE International Conference on Automation Science and Engineering (2015). https://doi.org/10.1109/coase.2015.7294269
6. Luo, J., Solowjow, E., Wen, C., Ojea, J.A., Agogino, A.M., Tamar, A., Abbeel, P.: Reinforcement learning on variable impedance controller for high-precision robotic assembly. In: 2019 International Conference on Robotics and Automation, Montreal (2019)
7. Mason, M.T.: Toward Robotic Manipulation. Ann. Rev. Control Robot. Auton. Syst. **1**(1), 1–28 (2018). https://doi.org/10.1146/annurev-control-060117-104848
8. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017)
9. Siqueira, A., Terra, M.: Nonlinear and Markovian H∞ controls of underactuated manipulators. IEEE Trans. Control Syst. Technol. **12**(6), 811–826 (2004). https://doi.org/10.1109/tcst.2004.833626
10. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction, 2 edn. A Bradford Book (2018)
11. Taylor, M.E., Stone, P.: Transfer learning for reinforcement learning domains: A survey. J. Mach. Learn. Res. **10**(1), 1633–1685 (2009)