Reinforcement Learning for Robotic Assembly with Force Control

Jianlan Luo
Pieter Abbeel, Ed.

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2020-20
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2020/EECS-2020-20.html

February 26, 2020

Acknowledgement

I am deeply grateful to my research advisor Professor Pieter Abbeel for his tremendous support; Pieter is always encouraging and pointing to the right research direction. This thesis would not be possible at all without Pieter's continuous engagement. I would also express my deep thanks to Professor Aviv Tamar who was then a post-doc scholar at Berkeley; for his mentorship, patience and time revising this thesis. I would also thank Professor Alice Agogino for her time serving on the second reader of this thesis. Some of results presented in this paper was supported in part by Siemens.

Reinforcement Learning for Robotic Assembly with Force Control

by Jianlan Luo

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

Professor Pieter Abbeel
Research Advisor

15-FEB-2020

(Date)

Professor Alice Agogino
Second Reader

15-FEB-2020

(Date)

Acknowledgments

I am deeply grateful to my research advisor Professor Pieter Abbeel for his tremendous support; Pieter is always encouraging and pointing to the right research direction. This thesis would not be possible at all without Pieter's continuous engagement. I would also express my deep thanks to Professor Aviv Tamar who was then a post-doc scholar at Berkeley; for his mentorship, patience and time revising this thesis. I would also thank Professor Alice Agogino for her time serving on the second reader of this thesis. Some of results presented in this paper was supported in part by Siemens.

Abstract

Reinforcement Learning for Robotic Assembly with Force Control

by

Jianlan Luo
Master of Science in Computer Sciences

University of California, Berkeley
Professor Pieter Abbeel, Chair

Precise robotic manipulation skills are desirable in many industrial settings, reinforcement learning (RL) methods hold the promise of acquiring these skills autonomously. In this paper, we explicitly consider incorporating operational space force/torque information into reinforcement learning; this is motivated by humans heuristically mapping perceived forces to control actions, which results in completing high-precision tasks in a fairly easy manner. Our approach combines RL with force/torque information by incorporating a proper operational space force controller; where we also explore different ablations on processing this information. Our method can be used in both inherently compliant and non-compliant robots; we tackle two specific use-cases: 1)deformable object manipulation 2)the open-source Siemens Robot Learning Challenge; both of them requires precise and delicate force-controlled robotic behavior. Video results are available at: https://sites.google.com/berkeley.edu/rl-robotic-assembly/

Contents

Chapter 1

Introduction

Today, industrial robots deployed across various industries are mostly doing repetitive tasks. The overall task performance hinges on the accuracy of their controllers to track pre-defined trajectories. To this end, endowing these machines with a greater level of intelligence to autonomously acquire skills is desirable. The main challenge is to design adaptable, yet robust, control algorithms in the face of inherent difficulties in modeling all possible system behaviors and the necessity of behavior generalization. Reinforcement learning (RL) methods hold promises for solving such challenges, because they promise agents to learn behaviors through interaction with their surrounding environments and ideally generalize to new unseen scenarios [1, 2, 3, 4].

In this research, we aim to learn policies for robotic assembly in high-precision settings. Specifically, we tackle two such problems detailed in Chapter 2 and Chapter 3 respectively : 1) assembly of a rigid peg into a deformable hole, where the diameter of the hole is smaller than that of the peg; and the robot is non-compliant which only provides position and velocity control interface; 2) assembly of high-precision work-pieces, in this case robot is compliant, we can access torque control interface.

In both tasks, the required precision exceeds the robot position controller's accuracy. In real manufacturing, human labor can accomplish such high-accuracy complex tasks in a fairly easy manner. For example, a peg in hole insertion is achieved by "feeling" the contacts. This can be achieve with heuristics based on force feedback, for instance by probing the hole before inserting or moving the peg around the surface to search for the insertion point. Hence, designing robust strategies by properly processing observations is more desirable than estimating perfect physical dynamics. RL allows to find control policies automatically for problems where traditionally heuristicis have been used. The question arises *how do we properly integrate observed force information into reinforcement learning process to produce desirable behaviors?*

RL is a method for learning such reactive policies automatically, through trial and error interaction in the domain, guided only by a reward signal that specifies how well the robot is performing the task. In practice, RL requires an informative reward

signal to works effectively, which can be hard to design automatically. With sparse reward that just specifies successful task completion, RL is prone to getting stuck in local optima. However operational space control could mitigate this problem by specifying high-level goals in task-space. [5, 6, 7, 8]. One particular way to achieve this is to combine RL with a proper operational force controller, this has the benefit of implicitly shaping control actions so that policies only search the space where a "good" solution exists for our problems.

Furthermore, combing RL with such force controllers is highly relevant to industrial applications if we want such robots to learn contact-rich manipulation skills. Currently, most policy search algorithms for contact-rich assemblies are implemented on inherently compliant robot arms such as the PR2, the iiwa or Rethink Robotics' Sawyer [9, 10, 11]. These robot arms have either passive compliance through spring mechanism in motors or have the ability to measure and command joint torques. These properties enable safe physical interaction of the robot with its environment, and joint torque readings can be encoded as features in learning algorithms to describe contact situations. Unfortunately, this is only of limited use for industrial applications because industrial robots are in general not compliant and offer only velocity and position control, but no torque control. However, they can often be equipped with a wrist force-torque sensor. While this still does not provide the ability to command joint torques, it opens the possibility for admittance force-torque control in task space, which introduces a principled way for such non-compliant robots to learn contact-rich tasks.

The reminder of this paper is organized as follows: Chapter 2 describes proposed method; Chapter 3 studies a case where proposed method is applied to a non-compliant robot assembling deformable objects; Chapter 4 studies another case where a compliant robot needs to learn more complex assembly skills.

Chapter 2

Related Work and Proposed Method

2.1 Related Work

Recent advances in RL have gained great success in solving a variety of problems from playing video games [12, 13, 14] to robotic locomotion[15, 16, 17, 18, 19], manipulation[20, 21, 22, 23, 24, 25, 4, 26, 27, 28]. Reinforcement learning can be distinguished in model-based methods and model-free methods [1, 3, 2]. While model-based policy search is computationally more expensive than model-free methods, it requires less data to solve a task. Recent progresses in the area of Deep Neural Networks suggest to also deploy them for parametrizing policies and other functions in RL methods [4, 29, 17]. This is often referred to as Deep Reinforcement Learning (DRL).

A recently developed model-based reinforcement learning algorithm called guided policy search (GPS) provided new insights into training end-to-end policy for solving contact-rich manipulation problems [18, 29, 30, 4, 31]; however; this method is not suitable for this high-precision setting because it has no means of avoiding local optima by its formulation. There are also approaches tackling this problem by explicitly modeling contact dynamics [32, 33, 34, 35, 36] Inoue et al. [37] use LSTM to learn two separate policies for finding and inserting a peg into a hole; however, their methods require several pre-defined heuristics, and also the action space is discrete.

Thomas et al. [38] combine RL with a motion planner to shape state cost in high-precision settings. This method essentially learns a trajectory following torque controller, and assumes access to a trajectory planner that could roughly avoid local optima. They also encode such planned reference into a neural network with attention mechanism, they show good generalization results in simulation.

2.2 Preliminaries

We consider all tasks here that can be described as moving already-grasped objects to their goal position. This is the most common setting in today's manufacturing. The success of such tasks can be measured as minimizing the distance between objects and their goal positions. We make no particular assumptions about encountered dynamics during tasks especially contacts. These need to be learned by the robot from various interaction with its environment. Let $\mathbf{x}_t$ and $\mathbf{u}_t$ denote robot states and actions respectively ; $\ell(\mathbf{x}_t, \mathbf{u}_t)$ be the cost function related to the task, T be the time horizon of a task. Our problem can be formulated as

$$\min_{\mathbf{u}_1, \mathbf{u}_2 \ldots \mathbf{u}_T} \sum_{t=1}^{T} \ell(\mathbf{x}_t, \mathbf{u}_t)$$
$$s.t. \quad \mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t) \; t = 1, 2 \ldots T-1,$$

where f governs (unknown) system dynamics, $\mathbf{x}$, $\mathbf{u}$ can also be subject to other algebraic constraints.

We consider our control action $\mathbf{u}$ to be operational force controller $\mathcal{F}_{tip} = [F_x, F_y, F_z, M_x, M_y, M_z]$. They represent desired force/torque or impedance in operational space, our goal is to optimize them through reinforcement learning. We make no particular assumption about the implementation of such controller; neither the specific reinforcement learning algorithm.

2.3 Operational Space Force Controller

There are various ways to implement such controller in operational space with or without a wrist force/torque sensor [5] [39][40]. We present two such force controllers depending on whether a wrist force/torque sensor is available.

For setup in Figure 3.1, a wrist force/torque sensor is mounted on the robot's end-effector; we feed Cartesian-space force/torque sensor signals to the robot's Cartesian-space velocity control loop. Specifically, denote τ as the desired force/torque vector in tool space; this includes three forces and three torques in three corresponding Cartesian axes. $\hat{\tau}$ denotes the measured force/torque vector from the robot's wrist sensor. We apply a PD feedback controller to the difference between τ and $\hat{\tau}$, and feed the resulting control to the Cartesian-space velocity command interface of the robot. This involves calculating the inverse Jacobian matrix of the robot and converting Cartesian-space velocity into joint-space velocity. We set this admittance controller as output of reinforcement learning algorithm.

For setup in Figure 4.1, no force/torque sensor is available; however we have access to joint torque readings. We can convert operational space force into joint space using robot Jacobian, then close the control loop in joint space using joint

torque feedback. Let $\mathcal{F}_{tip}$ be the desired wrench on the end-effector, we can then express the control law in joint space as [41]

$$M(q)\ddot{q} + c(q,\dot{q})\dot{q} + g(q) + J^T(q)\mathcal{F}_{tip} = \tau, \tag{2.1}$$

where q represents joint angles in generalized coordinates, $M(q)$ is the inertia, $c(q,\dot{q})$ is the Coriolis matrix, $g(q)$ are gravitational forces, $J(q)$ is the Jacobian, τ is the torque vector applied to manipulators' joints. In many force control tasks, robots move slowly, hence we ignore acceleration and velocity terms in Eq. 2.1. For a 7-DOF Sawyer manipulator arm that we consider in this paper, we can also project torques to its non-empty nullspace. Denoting the nullspace torque vector as τ_{null}, joint space control law is:

$$\tau = g(q) + J^T(q)\mathcal{F}_{tip} + [I - J^T(q)J^{T\dagger}(q)]\tau_{null}, \tag{2.2}$$

where $J^{T\dagger}(q)$ is the pseudo-inverse of $J^T(q)$. The control law in Eq.2.2 is appealing and simple, but it would generate undesirable and dangerous motion without enough resistance provided by the environment. In our experimental setup, we do not assume in-contact situations of objects being assembled, there is a relative open free-space that the robot needs to move through; directly applying Eq.2.2 would result in continuous acceleration. To mitigate this issue, in all our experiments, we wrap a position loop with small gains around the controller in Eq.2.2:

$$\tau = g(q) + J^T(q)\mathcal{F}_{tip} + [I - J^T(q)J^{T\dagger}(q)]\tau_{null} \tag{2.3}$$
$$+ K_{qp}(q - q^*) + K_{qd}(\dot{q} - \dot{q}^*), \tag{2.4}$$

where K_{qp} and K_{qd} are diagonal gain matrices with small entries, q and $\dot{q}$ are current joint positions and velocities, q^* and $\dot{q}^*$ are the desired ones obtained via inverse kinematics from end-effector pose. Aforementioned $\mathcal{F}_{tip}$ will be calculated by an RL controller.

2.4 Model-based Reinforcement Learning algorithm: iLQG

The specific model-based reinforcement learning algorithm that we consider here is iterative Linear-Quadratic-Gaussian (iLQG) [42, 18]. It is sample efficient and convenient second-order methods are available to solve it quickly [43]. Let $\omega = \{\mathbf{x}_1, \mathbf{u}_1, \ldots, \mathbf{x}_T, \mathbf{u}_T\}$ denote a trajectory, such that $p(\omega) = p(\mathbf{x}_1)\prod_{t=1}^{T} p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)p(\mathbf{u}_t|\mathbf{x}_t)$, $\ell(\omega) = \sum_{t=1}^{T} \ell(\mathbf{x}_t, \mathbf{u}_t)$ denotes the cost along a single trajectory ω; where $\mathbf{x}$ typically consists of joint angles, end-effctor pose and their time derivatives. We wish to minimize this cost; the goal is to minimize the expectation $E_{p(\omega)}[\ell(\omega)]$ over trajectory

ω by iteratively optimizing linear-Gaussian controllers and re-fitting linear-Gaussian dynamics. This algorithm iteratively linearizes the dynamics around the current nominal trajectory, constructs a quadratic approximation of the cost, computes the optimal actions with respect to this approximation of the dynamics and cost by dynamic programming, and forward runs resulting actions to obtain a new nominal trajectory. We use subscripts, e. g. $\ell_{\mathbf{xu}t}$ to denote derivatives with respect to vector $[\mathbf{x}_t; \mathbf{u}_t]$. Under the dynamics model and the cost function described in this section, we can write the Q-function and value function as

$$\begin{aligned} V(\mathbf{x}_t) &= \frac{1}{2}\mathbf{x}_t^\intercal V_{\mathbf{x},\mathbf{x}t}\mathbf{x}_t + \mathbf{x}_t^\intercal V_{\mathbf{x}t} + \text{const} \\ Q(\mathbf{x}_t, \mathbf{u}_t) &= \frac{1}{2}[\mathbf{x}_t, \mathbf{u}_t]^\intercal Q_{\mathbf{xu},\mathbf{xu}t}[\mathbf{x}_t; \mathbf{u}_t] + [\mathbf{x}_t; \mathbf{u}_t]^\intercal Q_{\mathbf{xu}t} + \text{const}. \end{aligned}$$

We can solve for V and Q with a recurrence that can be computed backwards through time starting from the last time step $t = T$:

$$\begin{aligned} Q_{\mathbf{xu},\mathbf{xu}t} &= \ell_{\mathbf{xu},\mathbf{xu}t} + f_{\mathbf{xu}t}^\intercal V_{\mathbf{x},\mathbf{x}t+1} f_{\mathbf{xu}t} \\ Q_{\mathbf{xu}t} &= l_{\mathbf{xu}t} + f_{\mathbf{xu}t}^\intercal V_{\mathbf{x}t+1} \\ V_{\mathbf{x},\mathbf{x}t} &= Q_{\mathbf{x},\mathbf{x}t} - Q_{\mathbf{u},\mathbf{x}t}^\intercal Q_{\mathbf{u},\mathbf{u}t}^{-1} Q_{\mathbf{u},\mathbf{x}t} \\ V_{\mathbf{x}t} &= Q_{\mathbf{x}t} - Q_{\mathbf{u},\mathbf{x}t}^\intercal Q_{\mathbf{u},\mathbf{u}t}^{-1} Q_{\mathbf{u}t}. \end{aligned} \tag{2.5}$$

This actually leads to the MaxEntropy LQR objective whose solution is a time-varying linear-Gaussian controller $p(\mathbf{u}_t|\mathbf{x}_t)$:

$$\min_{p(\mathbf{u}_t|\mathbf{x}_t)} \sum_{t=1}^{T} E_{p(\mathbf{u}_t|\mathbf{x}_t)}[\ell(\mathbf{x}_t, \mathbf{u}_t) - \mathcal{H}(p(\mathbf{u}_t|\mathbf{x}_t))]$$

The entropy term in the objective function above encourages exploration to acquire diverse samples for fitting dynamics. This objective can be optimized by setting $p(\mathbf{u}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{K}_t\mathbf{x}_t + \mathbf{k}_t, \mathbf{C}_t)$, and $\mathbf{C}_t = Q_{\mathbf{u},\mathbf{u}t}^{-1}$.We brief the overall scheme in Alg. 1.

Algorithm 1 Force-based RL controllers

1: **for** iteration $k \in \{1, ..., K\}$ **do**
2: Train local RL controller using iLQG, where $\mathbf{u}_t$ is set as operational space force controller
3: Project calculated desired force to joint space according to Chapter 2.3 depending on robot type
4: **end for**

Chapter 3

Case 1: Non-compliant robot arm with deformable objects

3.1 Problem Statement

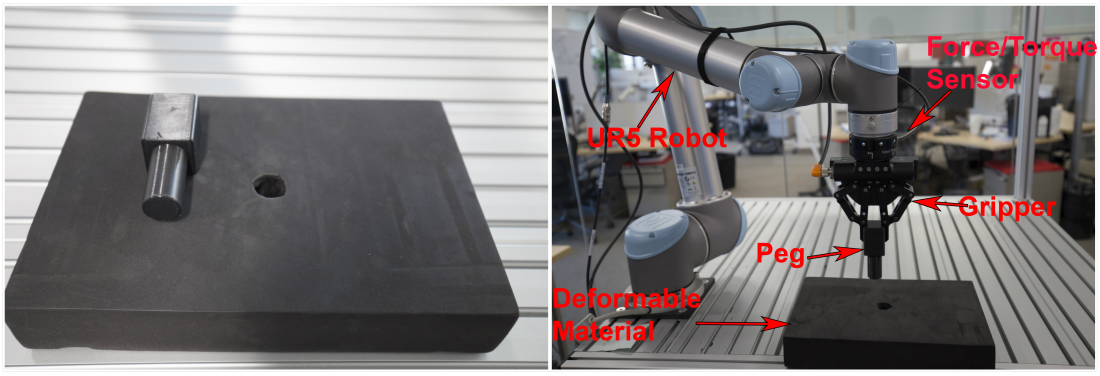


Figure 3.1: Experimental setup for peg insertion task.

Consider a robot arm without the possibility of joint torque sensing or joint torque control as in Fig. 3.1 The robot has no inherent compliance and permits only position and velocity control. However, the robot arm is equipped with a wrist mounted force-torque sensor, whose signals are available. Moreover, the joint angles of the robot are accessible as well. The robot end-effector holds a rigid peg whose position and orientation relative to the end-effector are known with a certain degree of uncertainty. Moreover, the grasp is not tight and the peg exhibits some bounded motion relative to the end-effector whenever the peg is in contact with the environment.

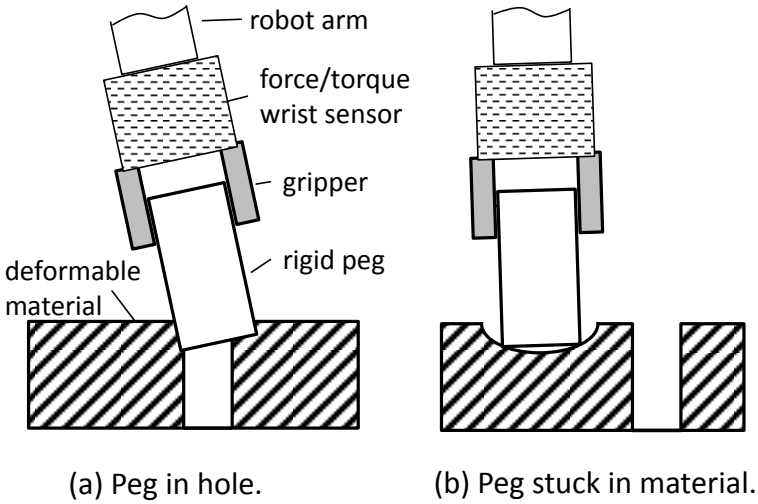


Figure 3.2: Assembly modes for rigid peg and deformable material.

A work piece with a hole is placed in the work space of the robot. The work piece consists of a deformable material with nonlinear material properties. The diameter of the hole is smaller than the diameter of the peg. The position of the whole relative to the robot frame is known, but also exhibits uncertainty.

The goal of the peg-in-hole problem is to control the robot in order to insert the rigid peg into the deformable hole. It is challenging to design a feedback control law for achieving this task due to the unknown contact mechanics. Figure 3.2 illustrates the assembly situation. In Figure 3.2 (a) the peg is successfully inserted into the elastic hole. The deformation posses challenges to the insertion tasks, because unlike rigid pairings, material yields if the robot pushes the peg into it as shown in Fig Fig. 3.2 (b). The key to escape from this local optimum is to properly adjust force/torque on the end-effector promptly.

3.2 Experiments

3.2.1 Experimental Setup Details

As shown in Figure 3.1, we use a UR5 robot manufactured by Universal Robots, Polyscope version 3.5 with the motherboard in the control box having been upgraded to CB3.1. We use the force/torque sensor FT300 from Robotiq with a range of measurements of 300 N for forces and 30 Nm for torques. The noise levels (estimated standard deviations) for F_x, F_y are 1.2 N; F_z is 0.5 N; M_x, M_y are 0.02 Nm; and M_z is 0.03 Nm. The data output frequency is 100 Hz. We use a 85mm adaptive gripper from Robotiq. We remove the fingertips from the gripper, and fixed the peg directly

to the gripper. However, the gripper connection allows bounded motions of the peg, which introduces some uncertainty. The hole is part of a work piece, which is made of ethylene-vinyl acetate. Its diameter is 20 mm while the peg's diameter is 25 mm. The cycle time for the UR5 is 8 ms, the robot does not provide access to its real-time control loop, also the force/torque sensor rate is 100 Hz. The robot neither has joint compliance nor does it offer joint torque control. Hence, it is well suited to benchmark our approach.

We vary the robot starting configuration for training local policies. The algorithm will be iteratively executed by deterministic reset to the same configuration after sampling trajectories. We collect five samples for each starting configuration in every iteration. We send control commands at 25 HZ; and each trajectory is 3.5 seconds long.

The robot state space consists of joint angles, joint velocities, the end-effector pose, which is represented by three Cartesian points and the velocities of these points. For an initial condition (robot configuration), we specify their target end-effector pose by three points in end-effector plane, the cost function is given by $r_\ell(d) = wd^2 + v\log(d^2 + \alpha)$, with $\alpha = 10^{-5}$, $v = 0.01$ and $w = 1.0$. This shaped cost function performs better than an l_2 distance cost function because the second term encourages precise placement near the target position. In order not to damage the material, in all our experiments, we locked rotations in three Cartesian axes, thus the policy outputs an action $\mathbf{u}_t = [F_x, F_y, F_z]$. Denote the forces measured by F/T sensor as $\hat{\mathbf{u}}_t$, the commanded joint velocity $\mathbf{v}_t$ to the robot can be calcuated as following:

$$K_p * (\mathbf{u}_t - \hat{\mathbf{u}}_t) = J_t * \mathbf{v}_t$$

where K_p is the feedback gain, and J_t represents robot Jacobian matrix.

3.2.2 Results

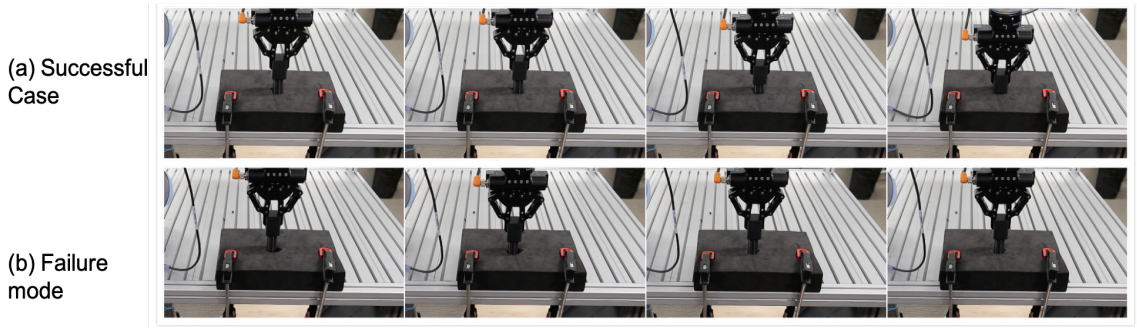


Figure 3.3: (a) Successful insertion. (b) Failure mode where the policy is unable to locate the hole.

We design our experiments by varying the starting positions of the peg, near to far; to test the if our method could learn policies effectively searching for the hole.

Since the robot can only be position or velocity controlled, 1) it is not capable of running RL algorithm using torque control, which will provide compliant behaviors; 2) it is not capable of running RL algorithm directly commanding velocity or position either, it will stick to pre-calculated trajectories thus yielding unexpected large force or moments when facing contact. So, we consider only one baseline that includes several way-points, and use robot position controller to track them; since the hole is smaller than the peg, this baseline almost failed every single time because of safety stop from the robot itself. We report following results in Figure 3.4, three different starting positions are tried, the distance is regarded as difference of horizontal projection between center of peg and hole; we can see the policy training is able to reach a reliable convergence in roughly five iterations. Success rate comparison of our method and aforementioned baseline is listed on Table 3.1.

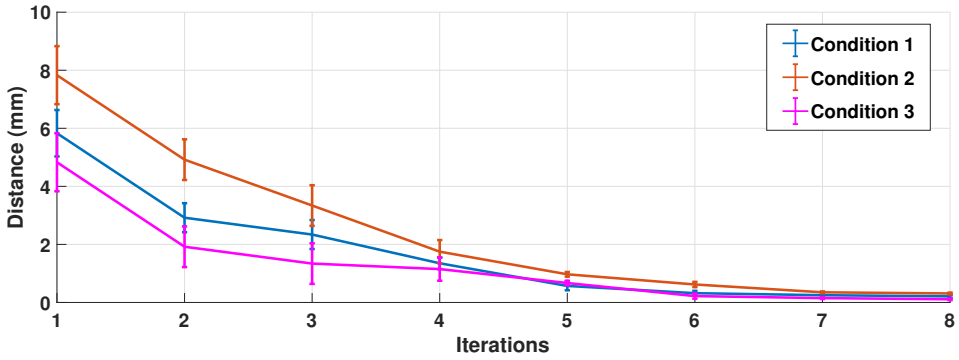


Figure 3.4: Learning curve in three experiments starting from different positions.

Table 3.1: Comparison of success rate, with different distance to the hole.

distance to hole	1 mm	6 mm	8 mm
our method	5/5	5/5	3/5
baseline	3/5	0/5	0/5

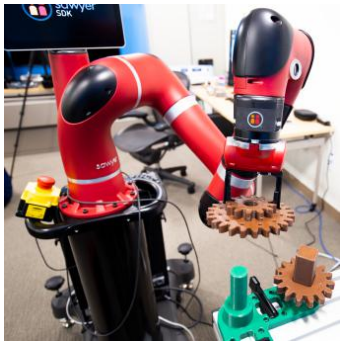
Figure 3.3(a)(b) shows some of the successes and failures. Several learned strategies can be observed. In the case that the initial position is slightly off the hole, the robot tends to apply large vertical forces to "slide" into the hole. While for the case that the initial position is further off the hole, the robot often first gets stuck in the material, then slightly lifts its arm to reduce the exerted force from the material, and moves its arm quickly after the lifting to find the hole.

To summarize our contributions: We introduce a principled way for extending non-compliant robots to learning contact-rich manipulation skills. Currently, most policy search algorithms for contact-rich assemblies are implemented on inherently compliant robot arms such as the PR2, the iiwa or Rethink Robotics' Sawyer

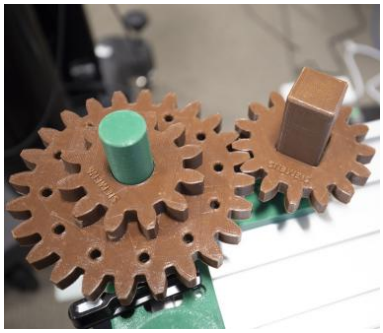
[9, 10, 11]. These robot arms have either passive compliance through spring mechanism in motors or have the ability to measure and command joint torques. These properties enable safe physical interaction of the robot with its environment, and joint torque readings can be encoded as features in learning algorithms to describe contact situations. Unfortunately, this is only of limited use for industrial applications because industrial robots are in general not compliant and offer only velocity and position control, but no torque control. However, they can often be equipped with a wrist force-torque sensor. While this still does not provide the ability to command joint torques, it opens the possibility for admittance force-torque control in task space. We show how this can be exploited for GPS usage even if joint torques cannot be directly commanded, but only positions and velocities. To the best of our knowledge force-torque signals have not been incorporated in variants of the GPS algorithm yet.

Chapter 4

Case 2: Compliant robot arm with rigid objects



a) Robot learning for complex assemblies.

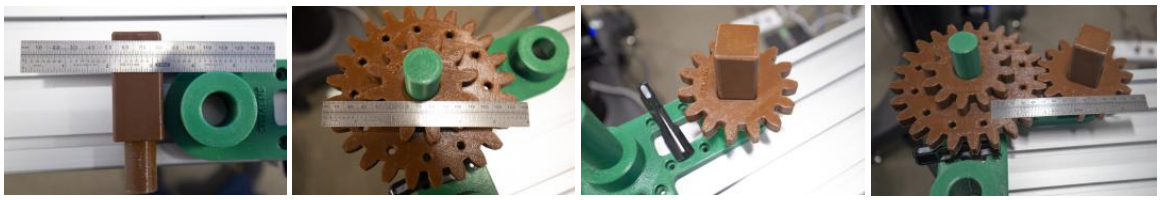


b) Assembled gear.

Figure 4.1: Learning control policies for assembly tasks.

4.1 Problem Statement

Consider the task of assembling the gear set shown in Fig. 4.1. The gear model was introduced by Siemens Corporation as a benchmark task for robotic assembly www.usa.siemens.com/robot-learning. The overall assembly task consists of four sequential steps, which are illustrated in Fig.4.2: first the robot needs to insert a cylindrical peg into its matching hole; then the large brown gear should be inserted through the gear shaft; then the small brown gear with the squared hole should be assembled; lastly the gear wheels need to be matched by aligning the corresponding gear teeth. In general the tolerances are tight. For example, step two requires tolerances tighter than 0.1 mm, which is beyond most deployed industrial robots' accuracy



a) Round peg in round hole. b) Gear wheel on shaft. c) Squared hole on squared shaft. d) Teeth Alignment.

Figure 4.2: Four tasks that represent different assembly challenges. Each task requires a flexible control policy that needs to consider contacts and friction.

today. Additionally, in step two, the peg can freely rotate at contact, the gear must be precisely oriented to match the squared peg; in step three, the small brown gear must be rotated by the large brown gear properly so that they can align with each other. This poses additional challenges: since none of these pegs or gears are fixed during assembly, this added uncertainty makes assembly even more difficult.

4.2 Experiments

In this section we answer the following questions. How does the proposed iLQG with force control perform? Is it actively exploiting contact constraint dynamics as we hypothesized? How does it compare to its ablations where force information is integrated differently?

4.2.1 Experimental Setup Details

We evaluate our methods on four assembly tasks, which are shown in Fig.4.2. We use a Rethink Robotics Sawyer robot. Sawyer offers an interface to query its wrist force/torque measurement, the noise levels (estimated standard deviations) for F_x, F_y are 2.0 N; F_z is 0.5 N; M_x, M_y are 0.5 Nm; and M_z is 0.1 Nm. Sawyer is commanded via ROS at 20 Hz. During training, we take four roll-outs per iteration. Typically, it takes three iterations to achieve successful behaviors, five iterations for convergence. We define a plane by three points in end-effector space, the cost function is a weighted mixture of the ℓ_1 and ℓ_2 norms of the differences between the current plane and the target plane as specified by the three aforementioned points.

4.2.2 Assembly Performance Results

We compare our method with the following baselines:

- **Kinematics Only:** We verify the difficulty of the tasks and the necessity for learning as a sanity check. For task 1 and task 2, we only specify target poses; for the more difficult task 3 and task 4, we also introduce several way-points.

Note that for task 1 and 2, we should get the same result every single time since robot kinematic controller is deterministic as well as these tasks. But for task 3 and 2, the peg and gear can move freely, so it is hard to specify the desired trajectory. For all these tasks, we use built-in position controllers from Sawyer robot.

- **iLQG with torque control:** This is the main baseline for comparison. The control actions from iLQG are directly the seven joint torques. For comparison, we use the same cost function as in our method, i.e., sparsely-defined target end-effector pose, no intermediate way-points are introduced.

- **iLQG with torque control, augmented state space:** We augment the state space with the F/T vector such that $\tilde{\mathbf{x}}_t = [\mathbf{x}_t, f_t]$, where f_t are F/T measurements. We apply direct torque control. The purpose of this is to verify if other formulation other than what we proposed could also actively use this additional information.

- **Our Method:** We refer to the operational space controller in Chapter 2 with iLQG. Sawyer robot provides an interface to directly command desired operational space forces, we use this for experiments where we only use these forces as control; we implement our own controller in eq.2.3 where position loop is wrapped.

- **Our Method with augmented state space:** Additional to operational space controller, we augment the state space to $\tilde{\mathbf{x}}_t = [\mathbf{x}_t, f_t]$. This experiment is for verifying if our method can be further improved.

A success is considered if an object is being assembled to a desired pose with defined tolerance. We report success rates for each individual task separately, because we train an individual policy for each task. However, it would be straightforward to report overall success rate by multiplying individual success rates together since policies are trained independently. We execute learned policies after training to calculate the success rates. Table 4.1 presents aforementioned success rates for four different tasks.

We interpret these results several fold: (1) kinematics baseline fails consecutively, this confirms the required accuracy and complexity for the gear set; (2) a vanilla iLQG with torque control, but without extensive cost shaping fails; the single success we observed is due to Gaussian noise in the controller, which generated some lucky motion to insert, and it is on the easiest task. (3) we did not find reliable improvement by augmenting state space with F/T information. Since F/T signals are not Markovian: F/T information at time step t τ_t is not necessarily a function of previous time step τ_{t-1}. Therefore fitting a time-correlated dynamics model to them does not produce meaningful information. We made several interesting observations during the experiments. During task one, the robot moves quickly in free-space to reach in-contact status, then it reduces its speed to slowly probe around, trying to

Table 4.1: Comparison of success rates for different tasks. Baseline 1 refers to kinematics only; baseline 2 refers to iLQG with direct torque control; baseline 3 refers to iLQG with direct torque control, augmented state space, our method w/ augmented refers augmented state space in our method.

	Task 1	Task 2	Task 3	Task 4
baseline 1	0/5	0/5	0/5	0/5
baseline 2	1/5	0/5	0/5	0/5
baseline 3	0/5	0/5	0/5	0/5
our method	**5/5**	**5/5**	2/5	**4/5**
our method w/ augmented	5/5	5/5	**3/5**	3/5

"feel" the surface; once it has a level of confidence of the hole's position, it becomes aggressive towards the goal it predicted, resulting in quick motions followed by a large downward force to complete insertion. The most interesting experiment is task 3, where the added uncertainty comes from a rotating peg. The robot first brings the small gear in contact with the peg, while applying a downward force so that small gear would not fall into free-space again; but this amount of downward force also allows room for applying additional rotating torque to the peg and gear aligning them with each other roughly; then the downward force increases to try insertion, if not successful, downward forces decrease but small horizontal force are also observed to fine-tune poses, this procedure iterates until the peg is fully inserted. This kind of behavior roughly aligns with humans' heuristics when facing such tasks. We also notice that the generated noise has some effect on the final performance, but not dominant. For instance, in task 4 where one gear needs to rotate to match the other one; if we remove the noise term from Gaussian policy, it will harm the performance a little bit; however as the learning process goes on, the learned rotating torque will becomes larger, thus compensate the performance loss of removing exploration noise.

Fig.4.3 shows computed actions at nearly convergence (only desired forces in x-directions and y-directions are plotted) for task 2 during one successful insertion. It is interesting to observe how the variance computed by the policy changes over time: initially, there is a certain level of variance for exploration to search for the target position; once the policy is confident about the goal, the variance reduces dramatically, the robot aggressively moves the object towards the goal; finally during the insertion phase, a certain level of noise is again injected for fine-tuning the gear's pose to overcome friction. This force-based insertion pattern is automatically discovered through interactions by the algorithm, and matches a human's intuition on such tasks well. Fig.4.4 presents F/T measurements during a successful insertion. We can observe peaks in the data, which motivates explicit use of F/T measurements, because of its

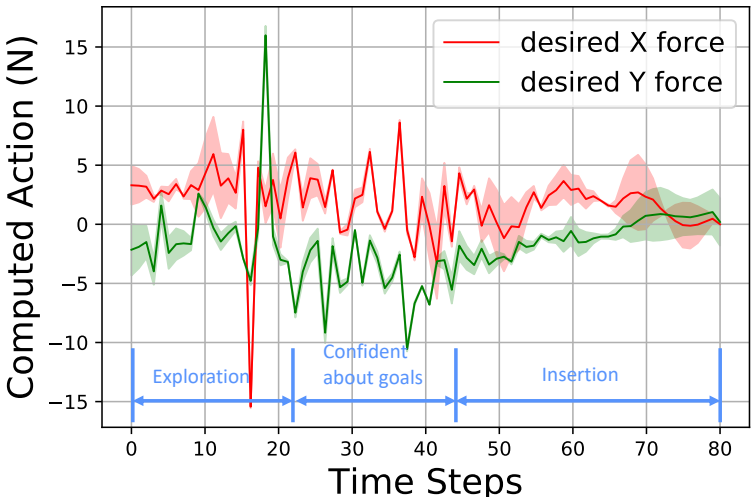


Figure 4.3: Action computed by learned policy during one successful insertion. Solid line shows computed action mean, and error bar for computed variance.

informative nature.

4.3 Combination of Learned Policy with Grasping

In all previous discussions, we assume objects are pre-grasped by the robot, and firmly held in the gripper. Here, we show the learned policy could actually be combined with a grasping pipeline; so that objects could be picked up anywhere but always moved to the position it was trained; this is useful for industrial automation applications since pre-grasp is not always available. Some uncertainty is also introduced to the policy since the grasping point could be different from where it was trained from. Figure 4.5 shows setup for this: 3D scanner (Photoneo) generates point clouds of scanned objects, then compare the objects with their CAD models to segment them out and perform pose estimation by surface matching. Once we know objects' poses, we grasp them at pre-selected points, then objects will be moved to positions where assembly policies were trained from; we run trained policies from there. Figure 4.6 shows scanned result and pose estimation results. This task consists of four sequential steps: first the robot needs to grasp the big gear and perform insertion using learned policy; then small gear needs to be grasped and insert into the gear base in the presence of big gear. Several way-points are added to smooth robot's motion; we use robot's motion controller to move between these way-points. We consider a success as all parts being picked up and assembled properly, we achieve 93.3% success rate (14/15), where the only failure resulted from small gear got stuck while being assembled into the big gear. A demo video is available on website.

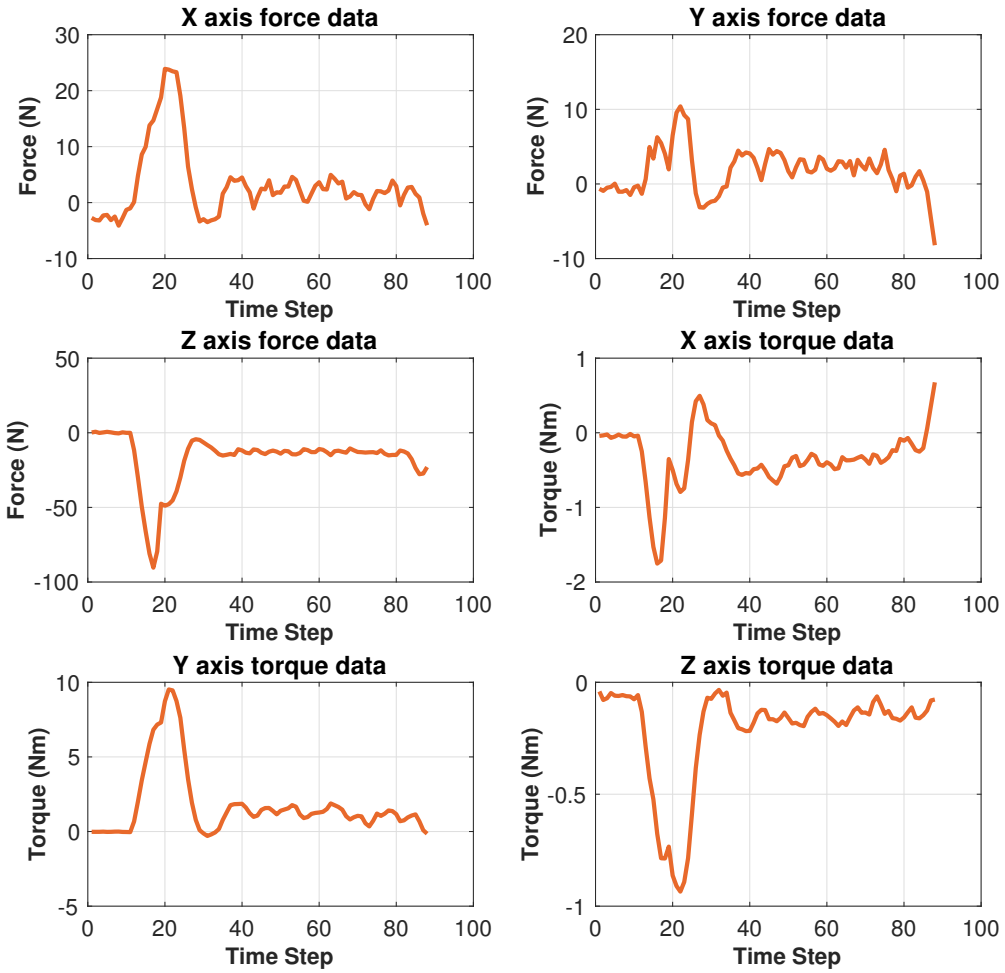


Figure 4.4: Six degree of freedom force torque measurements from a successful Task 2 insertion.

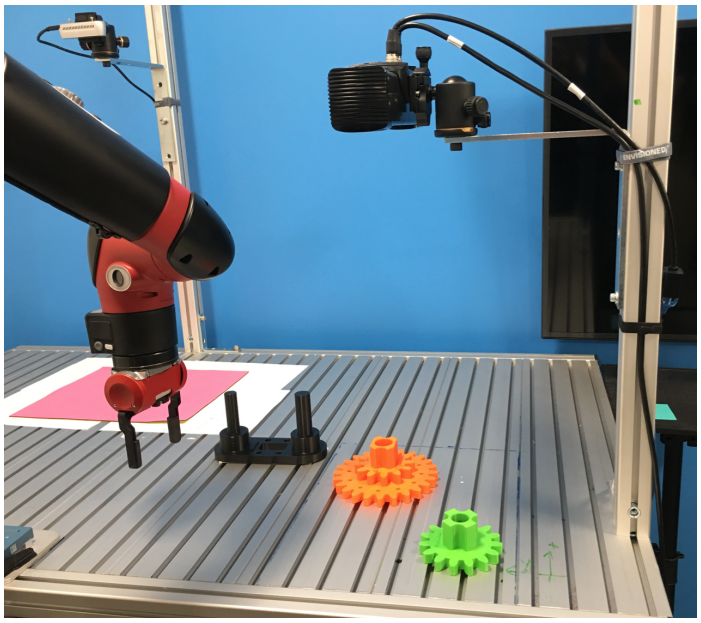


Figure 4.5: Setup for grasping and learned policy execution. A 3D scanner is mounted on the table, and registered to the robot frame. Objects are being placed on the table being scanned by the scanner. Gear base is fixed to the table.

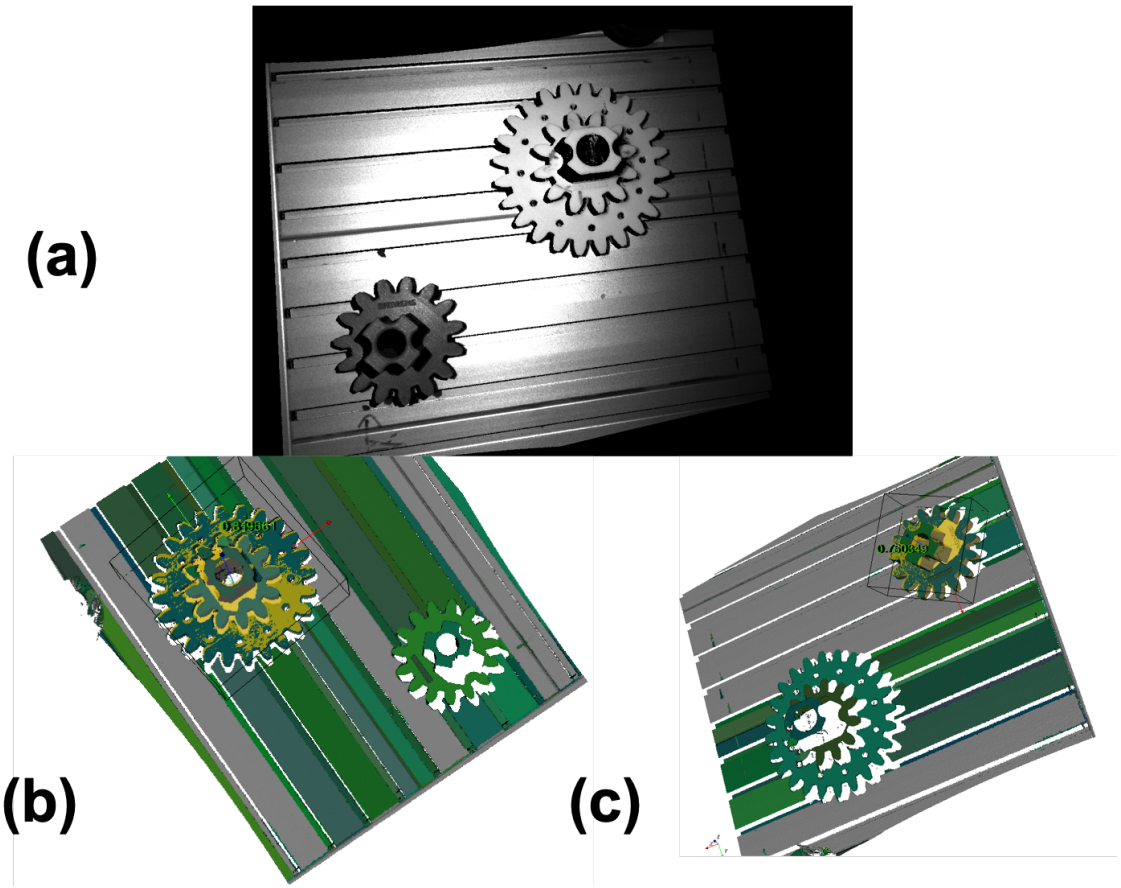


Figure 4.6: (a) Original scanned point cloud by scanner (b)Big gear segmentation and pose estimation (c)Small gear segmentation and pose estimation.

Chapter 5

Conclusion

In this paper we combine RL with an operational space force controller to solve the problem of high-precision robotic assembly. We specifically exploited one of the model-based RL algorithm, iLQG, compared with several ablations. We evaluated our method with two use-cases: deformable object manipulation with a non-compliant robot, high-precision assembly with a compliant robot. Results show our method can not only perform well in high-precision settings, but offer a principled way for industrial non-compliant robot learning contact-rich assembly skills.

Bibliography

[1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998, vol. 1.

[2] M. P. Deisenroth, G. Neumann, J. Peters *et al.*, "A survey on policy search for robotics," *Foundations and Trends in Robotics*, vol. 2, no. 1–2, pp. 1–142, 2013.

[3] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[4] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[5] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 1, pp. 43–53, February 1987.

[6] J. Peters and S. Schaal, "Learning to control in operational space," *The International Journal of Robotics Research*, vol. 27, no. 2, pp. 197–212, 2008.

[7] J. Nakanishi, R. Cory, M. Mistry, J. Peters, and S. Schaal, "Operational space control: A theoretical and empirical comparison," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 737–757, 2008.

[8] J. Buchli, F. Stulp, E. Theodorou, and S. Schaal, "Learning variable impedance control," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 820–833, 2011.

[9] Kuka , "Kuka lbr iiwa." [Online]. Available: https://www.kuka.com/en-us/products/robotics-systems/industrial-robots/lbr-iiwa

[10] Rethink Robotics, "Sawyer user guide." [Online]. Available: http://mfg.rethinkrobotics.com/mfg-mediawiki-1.22.2/images/1/1a/Sawyer_User_Guide_3.3.pdf

[11] Willow Garage, "Pr2 robot manual." [Online]. Available: https://www.clearpathrobotics.com/wp-content/uploads/2014/08/pr2_manual_r321.pdf

[12] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[13] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International Conference on Machine Learning*, 2016, pp. 1928–1937.

[14] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[15] P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng, "An application of reinforcement learning to aerobatic helicopter flight," in *Proceedings of the 19th International Conference on Neural Information Processing Systems*, ser. NIPS'06. Cambridge, MA, USA: MIT Press, 2006, pp. 1–8. [Online]. Available: http://dl.acm.org/citation.cfm?id=2976456.2976457

[16] J. Luo, R. Edmunds, F. Rice, and M. Agogino, "Tensegrity robot locomotion under limited sensory inputs via deep reinforcement learning," in *Robotics and Automation (ICRA), 2018 IEEE International Conference on*. IEEE, 2018.

[17] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International Conference on Machine Learning*, 2015, pp. 1889–1897.

[18] S. Levine and P. Abbeel, "Learning neural network policies with guided policy search under unknown dynamics," in *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[19] T. Zhang, G. Kahn, S. Levine, and P. Abbeel, "Learning deep control policies for autonomous aerial vehicles with mpc-guided policy search," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 528–535.

[20] S. Levine, N. Wagener, and P. Abbeel, "Learning contact-rich manipulation skills with guided policy search," in *International Conference on Robotics and Automation (ICRA)*, 2015.

[21] Y. Chebotar, M. Kalakrishnan, A. Yahya, A. Li, S. Schaal, and S. Levine, "Path integral guided policy search," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 3381–3388.

[22] J. Peters, K. Mülling, and Y. Altun, "Relative entropy policy search." in *AAAI*. Atlanta, 2010, pp. 1607–1612.

[23] J. Peters and S. Schaal, "Reinforcement learning of motor skills with policy gradients," *Neural networks*, vol. 21, no. 4, pp. 682–697, 2008.

[24] J. Fu, S. Levine, and P. Abbeel, "One-shot learning of manipulation skills with online dynamics adaptation and neural network priors," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE, 2016, pp. 4019–4026.

[25] V. Kumar, E. Todorov, and S. Levine, "Optimal control with learned local models: Application to dexterous manipulation," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 378–383.

[26] A. Tamar, G. Thomas, T. Zhang, S. Levine, and P. Abbeel, "Learning from the Hindsight Plan – Episodic MPC Improvement," *ArXiv e-prints*, Sep. 2016.

[27] P. Englert and M. Toussaint, "Learning manipulation skills from a single demonstration," *The International Journal of Robotics Research*, vol. 37, no. 1, pp. 137–154, 2018.

[28] I. Lenz, R. A. Knepper, and A. Saxena, "Deepmpc: Learning deep latent features for model predictive control," in *Robotics: Science and Systems*, 2015.

[29] S. Levine and V. Koltun, "Guided policy search," in *International Conference on Machine Learning*, 2013, pp. 1–9.

[30] W. H. Montgomery and S. Levine, "Guided policy search via approximate mirror descent," in *Advances in Neural Information Processing Systems*, 2016, pp. 4008–4016.

[31] Y. Chebotar, K. Hausman, M. Zhang, G. Sukhatme, S. Schaal, and S. Levine, "Combining model-based and model-free updates for trajectory-centric reinforcement learning," in *International Conference on Machine Learning (ICML) 2017*, Aug. 2017.

[32] S. S. M. Salehian and A. Billard, "A dynamical-system-based approach for controlling robotic manipulators during noncontact/contact transitions," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2738–2745, Oct 2018.

[33] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with gaussian mixture models," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, Oct 2011.

[34] I. Mordatch, E. Todorov, and Z. Popović, "Discovery of complex behaviors through contact-invariant optimization," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 43:1–43:8, Jul. 2012.

[35] I. Mordatch, K. Lowrey, and E. Todorov, "Ensemble-cio: Full-body dynamic motion planning that transfers to physical humanoids," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2015, pp. 5307–5314.

[36] I. Mordatch, K. Lowrey, G. Andrew, Z. Popovic, and E. V. Todorov, "Interactive control of diverse complex characters with neural networks," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 3132–3140.

[37] T. Inoue, G. De Magistris, A. Munawar, T. Yokoya, and R. Tachibana, "Deep reinforcement learning for high precision assembly tasks," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2017, pp. 819–825.

[38] G. Thomas, M. Chien, A. Tamar, J. Aparicio Ojea, and P. Abbeel, "Learning Robotic Assembly from CAD," *ArXiv e-prints*, 2018.

[39] J. Nakanishi, R. Cory, M. Mistry, J. Peters, and S. Schaal, "Operational space control: A theoretical and empirical comparison," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 737–757, 2008.

[40] B. Siciliano and L. Villani, *Robot force control*. Springer Science & Business Media, 2012, vol. 540.

[41] K. M. Lynch and F. C. Park, *Modern Robotics: Mechanics, Planning, and Control*, 1st ed. New York, NY, USA: Cambridge University Press, 2017.

[42] B. D. Ziebart, J. A. Bagnell, and A. K. Dey, "Modeling interaction via the principle of maximum causal entropy," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10. USA: Omnipress, 2010, pp. 1255–1262.

[43] E. Todorov and W. Li, "A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems," in *Proceedings of the 2005, American Control Conference, 2005.*, June 2005, pp. 300–306 vol. 1.