Appendices

A Task Setup

A.1 Simulation Details

We build the simulation environment using MuJoCo [1] simulation for learning robot contact-rich manipulation skills. In the simulation, we include the model of the FANUC LRMate 200iD robot, and each of the joints is controlled with motor torque command. We incorporated an F/T sensor asset on the robot's wrist to measure the contact force. For the low-level controller, we employed computed torque control [2] to track the compliant trajectory x_c and $\dot{x}_c$ derived from the admittance controller. The simulation time step was set to $0.01\ s$. Further details regarding the assembly and pivoting setups are outlined below:

Assembly: This task involves aligning a square-shaped peg with a hole. The edge length of the peg is $4\ cm$, and there is a clearance of $2\ mm$ between the peg and hole. The friction coefficient between the peg and hole is configured as 0.3.

Pivoting: In this task, the objective is to reorient a rectangular object against a rigid wall. The simulated object has dimensions of $10 \times 10 \times 2.6\ cm^3$. A friction coefficient of 0.7 is assigned to all objects in the simulation.

A.2 Real Robot Experiment Setup

The real robot setup is visualized in Fig. 1. We utilized FANUC LRMate 200iD industrial robot as the test bed for our real-world experiments. The end-effector pose, and velocity are obtained from the joint encoders. The end-effector pose, and velocity are obtained from forward kinematics. The contact force is measured by an ATI Mini45 Force/Torque sensor mounted on the robot's wrist. The low-level position/velocity controller is achieved via a Positional-Integral (PI) control law with feed-forward terms to cancel gravity and friction. The controller is implemented in Matlab Simulink Real-Time and runs on $1KHz$. The admittance controller we use takes in the desired robot motion x_d and optimized admittance control parameters P and outputs the command robot motion x_c to the low-level position/velocity controller. The robot motion x_d is directly sent from an Ubuntu computer with a User Datagram Protocol(UDP) in $125Hz$. Similarly, the initial control parameters P are sent from the Ubuntu computer and optimized in MATLAB with a built-in SQP solver.

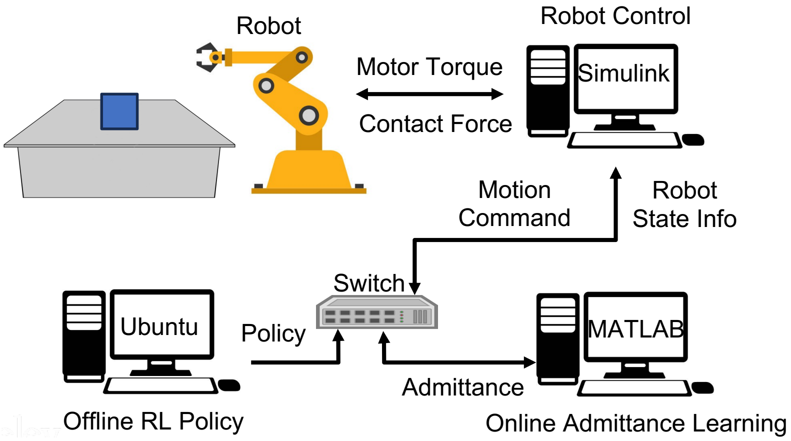

Figure 1: Real robot experiment setup.

B Simulation Training Details

B.1 Domain Randomization Details for Contact-rich Tasks

In both the assembly and pivoting tasks, we introduced Gaussian noise with a mean of zero and a standard deviation of $0.2\ N$ to the FT sensor readings as measurement noise. Additionally, we applied a clipping operation to the collected contact force, limiting it to the range of $\pm 10\ N$ for regulation purposes. To enhance the robustness of the learned skills, we incorporated randomization into the robot's initial pose.

For the assembly task, the robot's initial pose was uniformly sampled from a range of $[\pm 30\ mm, \pm 30\ mm, 30 \pm 5\ mm]$ along the X, Y, and Z axes, respectively. As for the pivoting task, the range for the initial pose was set to $[150 \pm 30\ mm, 5 \pm 5\ mm]$ along the X and Z axes relative to the rigid wall.

B.2 RL Training Details

We use the Soft Actor Critic [3] with implementation in RLkit [4] to learn robot manipulation skills in simulation. The hyperparameter selections are summarized in Table. 1.

Hyperparameters	Assembly	Pivoting
Learning rate - Policy	1e-3	1e-4
Learning rate - Q function	1e-4	3e-4
Networks	[128,128] MLP	[128,128] MLP
Batch size	4096	4096
Soft target update (τ)	5e-3	5e-3
Discount factor (γ)	0.95	0.9
Replay buffer size	1e6	1e6
max path length	20	40
eval steps per epoch	100	400
expl steps per epoch	500	2000

Table 1: Hyperparameters for RL training

C Discussion on Proposed Approach

C.1 Discussion on the Necessity of Learning the Compliance Control Parameters

We consider the manipulation policy for contact-rich manipulation tasks to contain a manipulation trajectory and the corresponding compliance control parameters.

The main difference between 'contact-rich' manipulation and regular manipulation tasks is how much force the robot exerts on the environment. The more force the robot applies, the more force it has to withstand. For contact-rich manipulation, the robot desired trajectory often has to penetrate the object with its end-effector to generate enough force for the task. For example, to wipe a table, the robot has to push its end-effector below the table surface. Since the robot is a rigid object, it needs a compliance controller to regulate its behavior and prevent potential damage. Compared to a position/velocity controller that might not need to tune the PID gains frequently, a compliance control is very sensitive [5] to the change of environment or task goals. It thus requires careful tuning of the parameters for each task. Therefore, for contact-rich manipulation, a suitable policy should be matched with the appropriate compliance control parameters to achieve the task smoothly.

C.2 Discussion on Approaches for Modeling Contact Force

A key component in our online admittance learning is the dynamics constraint, as shown below:

$$\dot{x} = \begin{bmatrix} \dot{e} \\ \ddot{e} \end{bmatrix} = f(x, F_{ext}, u) = \begin{bmatrix} \dot{e} \\ -M^{-1}D\dot{e} - M^{-1}Ke + M^{-1}F_{ext} \end{bmatrix} \tag{1}$$

where we want to regulate the future robot behavior based on the current robot state and the external force F_{ext}. In optimization, when we change the admittance parameter M, K, and D, the robot motion will change, and the external force that the environment gives to the robot will change as well. Thus, a robust way to model the external force F_{ext} is crucial in our online admittance learning.

To estimate or approximate the contact force in real time, we compare four approaches:

- *record & replay*: We record the force/torque from the most recent measurements within a time window and directly use the pre-recorded data as F_{ext} in the optimization.
- *hybrid impulse dynamics*: We use Eq. 1 with $F_{ext} = 0$ when there is no contact. For the contact, we model it implicitly as $M\dot{x}^- = \gamma M\dot{x}^+$, where $\dot{x}^-$ and $\dot{x}^+$ are the robot end-effector velocities before and after the contact. By online fitting the γ, we can optimize these hybrid dynamics to calculate the optimal parameters.
- *analytical contact model with online parameter fitting*: We model the contact explicitly using analytical models and fit the necessary parameters using online data, following [6, 7].
- *contact force fitting*: We fit a contact force model using online force sensor measurements.

However, the *hybrid impulse dynamics* approach is not suitable for our requirements. As shown in Fig. 2, the contact force profile in contact-rich manipulation indicates that the robot maintains contact with the environment for most of the time. Therefore, neglecting the entire contact process and modeling it implicitly is not appropriate for our applications.

Similarly, *analytical contact model with online parameter fitting* does not fit our scenarios either. Although it has been successful in some pivoting tasks, it relies on the quasi-static assumption that does not hold in our scenario. One of the main challenges of transferring the admittance parameters is to avoid the robot bouncing on the object. Moreover, the analytical model assumes point or sliding contact modes, which may be hard to generalize to different tasks, such as assembly.

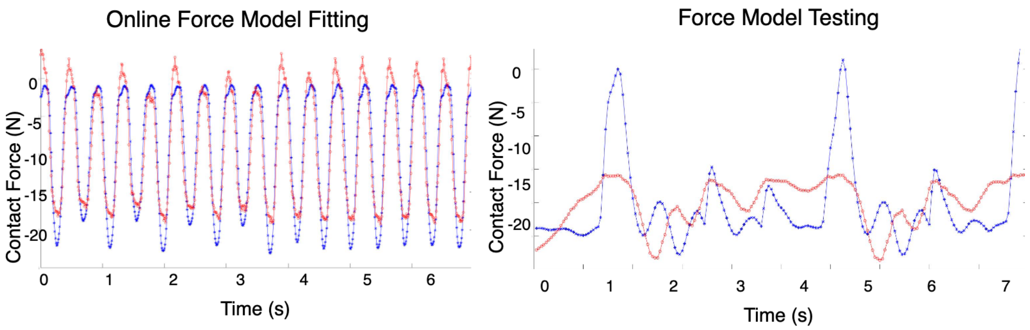

Figure 2: Performance of online force fitting (in z axis). In every time window, we collect the force/torque measurements and use the least square to fit the force model $F_{ext}(x, \dot{x}) = a(t)x(t) + b(t)\dot{x}(t) + c(t)$. On the left, it shows the linear model can fit the force profile locally. However, it can be extremely challenging to generalize to the next time window, as shown on the right.

Finally, for *contact force fitting*, we assume a linear (spring-damping) contact force model: $F_{ext} = a(t)x(t) + b(t)\dot{x}(t) + c(t)$ within a short time window. We use the least square to estimate the parameters a, b, and c in real time. Fig. 2 shows an example of fitting results. It can fit the force profile well in a short time window. However, as we need to apply the model learned in the previous time window to the next step, the generalization ability is poor as it is hard to capture the peak of the force profile. Experiment videos comparing the performance of *contact force fitting* and *record & replay* are available on our website. We can observe that the contact force fitting method cannot stabilize the robot during contact.

D Baseline Results

D.1 Sim-to-real Transfer

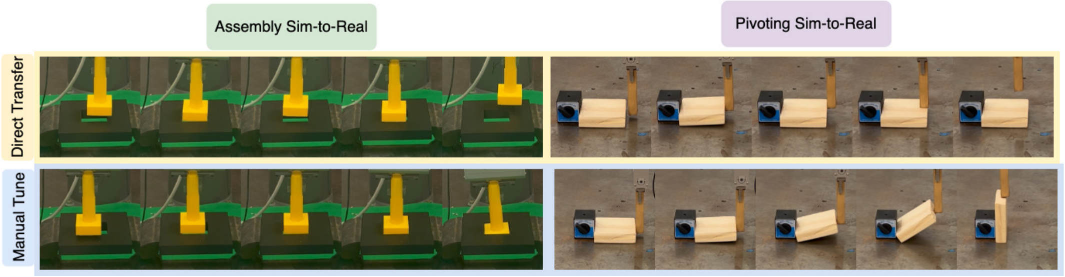

Figure 3: Snapshots of baseline approaches for the sim-to-real experiment. The control parameters learned in the simulation will result in a large contact force and makes the robot bounce on the surface, which will, in turn, result in failures of the tasks.

Here we provide snapshots of the baseline methods: *Direct Transfer* and *Manual Tune*. As introduced in the paper, *Direct Transfer* baseline utilizes the offline learned policy and directly applies it to the real robot without fine-tuning as [8] did. We hope the domain randomization on object position and force information can provide good generalizability and make it robust and transferable to real robots.

However, as shown in Fig. 3, direct applying the learned policy cannot achieve both tasks successfully. The main problem comes from the learned admittance control parameters. Where in the simulation, applying such parameters to the robot will not result in the robot bouncing on the object. In contrast, it can enable the robot to finish the task very efficiently. However, in the real world, such control parameters will result in large contact force and oscillation behaviors of the robot, which in turn, let the robot fails to establish stable contact with the object and finish the task.

For the *Manual Tune* baseline, we carefully tune the admittance parameters for each task in order to make the robot achieve smooth behavior during the contact. Table. 2 summarizes the parameters. As shown in Fig. 3, the manually tuned baseline can successfully achieve the task. However, since it requires human tuning, it is not time-consuming and task-dependent. A practical problem of manually tuning the control parameters is the need of trying various combinations of parameters. During this process, it is dangerous to let the robot interact with the environment and may cause damage to both the object and the robot.

Tuned Admittance Parameters	Assembly	Pivoting
End-effector Mass M (kg)	$[3, 3, 3]$	$[4, 4, 4]$
End-effector Inertia I (kgm^2)	$[2, 2, 2]$	$[2, 2, 2]$
Position Stiffness K (N/m)	$[200, 200, 200]$	$[300, 300, 300]$
Orientation Stiffness K (Nm/rad)	$[200, 200, 200]$	$[200, 200, 200]$
Position Damping D (Ns/m)	$[300, 300, 300]$	$[300, 300, 300]$
Orientation Damping D (Nms/rad)	$[250, 250, 250]$	$[250, 250, 250]$

Table 2: Manually tuned admittance control parameters for the experiments.

D.2 Generalization to Different Task Settings

In order to evaluate the generalization performance to different tasks, we conducted tests on various variations of tasks as depicted in Fig. 4. For assembly, these tasks included polygon-shaped peg holes, such as triangular peg-holes with an edge size of $51.4\ mm$ and a clearance of $1.4\ mm$, as well as pentagon peg-holes with an edge size of $57.8\ mm$ and a clearance of $1.3\ mm$. Additionally, we performed experiments on standard electric connectors, such as Ethernet and waterproof connectors, for further assessment.

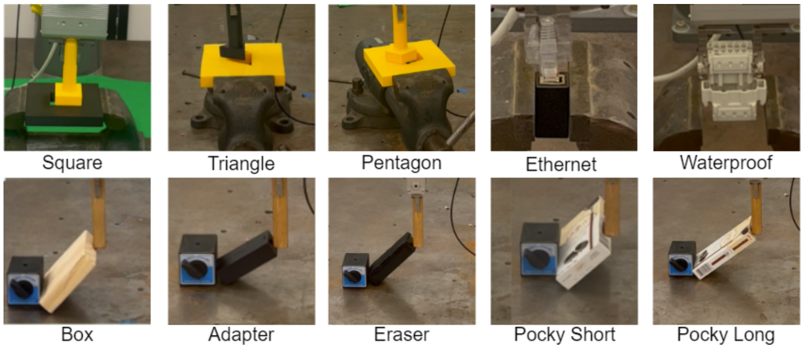

Figure 4: Real-world manipulation tasks

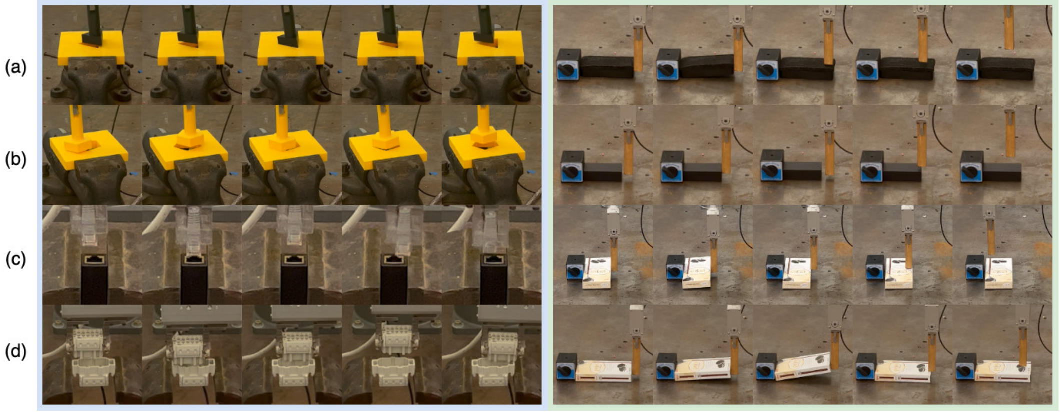

Figure 5: Snapshots of directly using the learned policy to generalize to various task settings. The snapshots and videos of the baseline methods are available on our website.

Regarding the pivoting task, we expanded the test set to include different objects. These objects consisted of an adapter with dimensions of $8.8 * 4.1 * 2.6\ cm^3$ and a weight of $69\ g$, an eraser with dimensions of $12.2 * 4.8 * 3.0\ cm^3$ and a weight of $36\ g$, and a pocky with dimensions of $14.8 * 7.9 * 2.3\ cm^3$ and a weight of $76\ g$.

The snapshots of the *Direct Transfer* and *Manual Tune* baselines can be seen in Fig.5 and 6, respectively. As observed in the sim-to-real experiments, the *Direct Transfer* baseline struggles to achieve stability during manipulation, resulting in failures when attempting to assemble or pivot objects of different shapes. On the other hand, the *Manual Tune* baseline demonstrates high success rates when dealing with polygon-shaped peg-holes and when pivoting the eraser. This success can be attributed to the similarity in geometric or dynamic properties between the learned object and these specific test objects. However, the *Manual Tune* baseline fails to generalize its performance to objects with significant differences, as illustrated in Fig.6(c) and (d).

E Current Limitations and Future Improvements

As we discussed in the paper, our current framework has three main limitations:

It assumes that the task settings in geometry are similar from training to testing. It uses a simple strategy for estimating the contact force. It has a relatively low update frequency and may not be suitable for manipulating fragile objects.

To address the first limitation, we plan to use meta-learning to learn the manipulation trajectory that can generalize well to different task settings. Meta-learning has been shown to be effective in generalizing the learned trajectory to various scenarios, and we believe that combining meta-

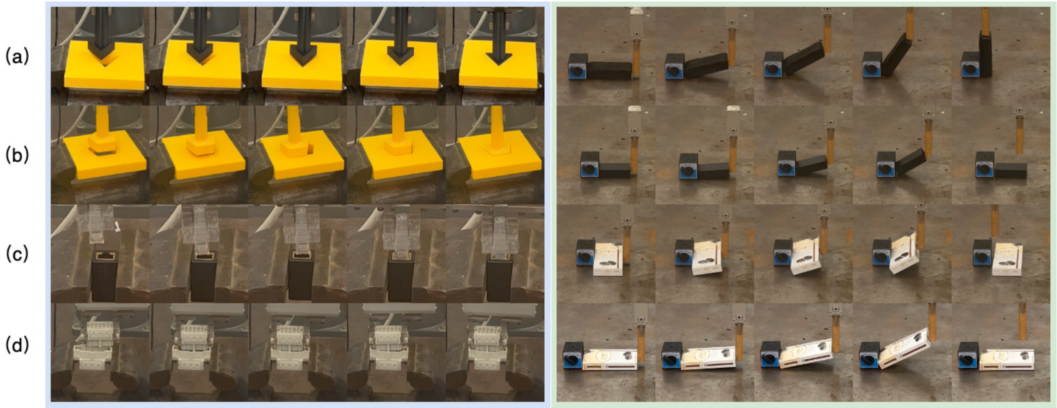

Figure 6: Snapshots of directly using learned trajectory and the manually tuned admittance control parameters to generalize to various task settings. The snapshots and videos of the baseline methods are available on our website.

learning and our proposed online residual admittance learning can bridge the sim-to-real gap for many contact-rich manipulation tasks.

For the second limitation, we are interested in exploring and experimenting with the analytical contact model approach as discussed in the Appendix. Using an analytical model and estimating the key parameters online may improve the performance. However, finding a general contact model or a method that can switch between different models will be the focus of our future work.

The last limitation is related to the time window size for online force/torque sensor data collection. We will try different time window sizes and increase the update frequency to enhance the adaptation performance in our future work.

References

[1] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ Int. Conf. on Intelligent Robots and Syst.*, pages 5026–5033. IEEE, 2012.

[2] J. J. Craig. *Introduction to robotics*. Pearson Educacion, 2006.

[3] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

[4] Rail-Berkeley. Rail-berkeley/rlkit: Collection of reinforcement learning algorithms. URL https://github.com/rail-berkeley/rlkit.

[5] C. Wang, X. Zhang, Z. Kuang, and M. Tomizuka. Safe online gain optimization for cartesian space variable impedance control. In *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*, pages 751–757. IEEE, 2022.

[6] N. Doshi, O. Taylor, and A. Rodriguez. Manipulation of unknown objects via contact configuration regulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2693–2699. IEEE, 2022.

[7] J. Zhou, M. T. Mason, R. Paolini, and D. Bagnell. A convex polynomial model for planar sliding mechanics: theory, application, and experimental validation. *The International Journal of Robotics Research*, 37(2-3):249–265, 2018.

[8] R. Martín-Martín, M. A. Lee, R. Gardner, S. Savarese, J. Bohg, and A. Garg. Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks. *arXiv preprint arXiv:1906.08880*, 2019.