Flow with the Force Field: Learning 3D Compliant Flow Matching Policies from Force and Demonstration-Guided Simulation Data

Tianyu Li[1*], Yihan Li[1*], Zizhe Zhang[1] and Nadia Figueroa[1]

***Abstract*— While visuomotor policy has made advancements in recent years, contact-rich tasks still remain a challenge. Robotic manipulation tasks that require continuous contact demand explicit handling of compliance and force. However, most visuomotor policies ignore compliance, overlooking the importance of physical interaction with the real world, often leading to excessive contact forces or fragile behavior under uncertainty. Introducing force information into vision-based imitation learning could help improve awareness of contacts, but could also require a lot of data to perform well. One remedy for data scarcity is to generate data in simulation, yet computationally taxing processes are required to generate data good enough not to suffer from the Sim2Real gap. In this work, we introduce a framework for generating force-informed data in simulation, instantiated by a single human demonstration, and show how coupling with a compliant policy improves the performance of a visuomotor policy learned from synthetic data. We validate our approach on real-robot tasks, including non-prehensile block flipping and a bi-manual object moving, where the learned policy exhibits reliable contact maintenance and adaptation to novel conditions. Project Website: flow-with-the-force-field.github.io.**

I. INTRODUCTION

Contact-rich robotic manipulation tasks demand a delicate balance between precise motion control and compliant force regulation. Mechanical compliance is crucial for successful contact interactions. Visuomotor policies, which are control policy representations that map raw visual observations to motor actions, have emerged as the leading robot learning paradigm for manipulation tasks due to their ease of specification and multi-modal capabilities. Yet, state-of-the-art approaches often ignore compliance and focus only on positional accuracy [1], [2], [3]. Recent advances have highlighted that incorporating compliance or force feedback can drastically improve performance in tasks like object flipping or wiping, where fixed-stiffness controllers fail to handle varying contact conditions [4], [5]. Some learn variable stiffness profiles from human demonstrations via diffusion models [4], [6], while others use reinforcement learning or trajectory optimization to tune compliance [7], [8].

While these methods achieve strong performance, they typically demand substantial human effort. For instance, ACP [4] requires hundreds of real demonstrations, and RL or trajectory optimization methods need carefully designed reward functions and extensive data or training. Consequently, current compliance policies lack scalability due to intensive physical data collection or intricate reward engineering. In contrast, our pipeline significantly reduces these burdens by generating force-informed simulation data from a single demonstration, potentially automating compliant policy learning for continuous, contact-rich tasks.

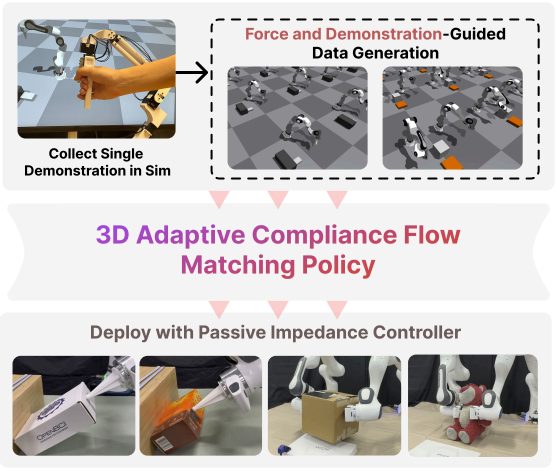


Fig. 1. Given *a single demonstration* in simulation, we generate force and demonstration-guided data and transfer to real-world compliant visuomotor policy deployment.

In this work, we address the challenge of learning vision-force adaptive compliance policies from simulation-generated data instantiated by a single human demonstration. The single demonstration is collected in a simulation environment via teleoperation [9], removing the need for a real robot setup. In particular, we propose a lightweight, yet effective, data generation strategy that produces diverse behaviors from a single demonstration in a simulator via force-informed trajectory modulation [10], [11], [4], [5] and Laplacian editing [12], [13], [14]. These synthetic trajectories are used to train an imitation policy that conditions on 3D pointcloud observations, similar to [15], [16], end effector pose, and force measurements. The policy outputs task-specific passive impedance parameters that can be executed by a low-level compliance controller on real robots. To learn this complex and multi-modal sensorimotor mapping, we leverage the flow-matching approach [17], [18], [3], allowing high-frequency inference. We introduce using point clouds and force input for the flow matching policy.

While a policy that outputs good trajectories is important, the rollout controller is equally critical for ensuring robustness and safety in the real world, especially under the Sim2Real gap. Thus, we encode the policy rollout poses into a state-velocity field, which, during execution, is coupled with a Passive Impedance Controller [19] that dampens

[1]Tianyu Li, Yihan Li, Zizhe Zhang, and Nadia Figueroa are with the GRASP Lab, University of Pennsylvania, PA 19104, USA.
[*] Equal contribution. Email: {tianyuli, yianlee}@seas.upenn.edu

deviations from the desired velocity. Unlike position-based controllers that rigidly follow waypoints and often generate abrupt contact forces, this velocity-based approach reduces energy injection with softer contact, as with state-dependent dynamical system (DS) policies [20]. Combined visuo-force as an input and the compliant vector as the output, this framework reduces the risk of damage due to misalignment while maintaining good performance.

To summarize, we propose a framework for learning adaptive compliant vision-based policies from simulation data, consisting of three major components, including i) generating force-informed sim data, ii) policy learning with flow matching, and iii) safe rollout on real hardware with state-velocity fields. Our contributions are fourfold:

- Propose a lightweight, yet effective, force-informed data generation strategy in simulation with virtual targets and Laplacian editing for vision-based imitation learning.
- We design an Adaptive Compliant Flow Matching policy that uses point cloud and force as inputs and outputs pose actions and impedance parameters.
- We demonstrate zero-shot transfer of our point cloud-based policy to real Franka robots on tasks like box flipping and bimanual grasping, without any real-world demonstrations or sim2real transfer algorithm.
- We design a scheme for generating a vector field from policy pose rollouts, enabling passive impedance controller to carry out the compliant policy on real robots with better performance and lower energy injection.

II. RELATED WORK

A. Learning Force-Informed Policies

Learning policies that integrate visual and force feedback with adaptive compliance has proven highly effective for contact-rich manipulation tasks. [4] proposed an Adaptive Compliance Policy with a diffusion model trained on human demonstrations, outperforming fixed-stiffness controllers. [5] introduced DexForce, augmenting kinesthetic demonstrations with force virtual targets to enhance dexterous task performance. FACTR [9] combined force-feedback teleoperation with curriculum training, encouraging the model to prioritize joint torque cues rather than solely vision. Our approach differs by employing point clouds instead of RGB inputs, enabling better spatial generalization and robustness to lighting variations. Moreover, rather than extensive real-world demonstrations, we generate synthetic, force-aware data in simulation from a single human demonstration. Reactive methods such as FoAR [21] and Reactive Diffusion Policy [22] integrate vision and force feedback to handle contact-rich tasks, yet still rely heavily on real or teleoperated data. Related approaches include ForceMimic [23], variable impedance learning [24], IRL-based methods for impedance recovery from demonstrations [25], and RL-based compliance approaches like Hybrid Trajectory and Force Learning (HTFL) [7]. These methods generally require extensive real-world demonstrations or involve intricate reward engineering and substantial simulation exploration. In contrast, our method learns adaptive compliance entirely from lightweight, simulation-generated data, eliminating the need for real-world demonstrations or expensive trial-and-error rollouts.

B. Data Generation for Sim2Real

Simulation-based data generation has become critical in reducing the costs and efforts associated with real-world demonstration collection. [26] introduced PhysicsGen, which generates large-scale datasets via trajectory optimization in simulation from limited human demonstrations, enabling zero-shot sim-to-real deployment. Similarly, [27] uses example-guided RL in simulation and distills policies for whole-body manipulation using state-based inputs and large amounts of simulated interaction. These systems require either carefully tuned optimization or RL parameters, and do not use rich input like point clouds and force. In contrast, our work introduces a direct trajectory warping technique for generating force-informed data for sim2real compliant policy with flow matching [28], [16] on point cloud and force inputs. Our lightweight method can provide more control over how to shape the trajectory for generating the data. In a same spirit as our approach, DemoGen [29] registers a single human demonstration with TAMP, synthesizing novel demonstrations by rearranging object configurations and adapting action trajectories via planning methods, enhancing spatial generalization. Nevertheless, DemoGen focuses on pick-and-place, object rearrangement tasks, not compliant continuous-contact driven tasks. Other methods also include force or contact cues in simulation, but each has limitations that our method addresses. Approaches such as FORGE [30], incorporate force information and domain randomization to train RL policies for generating data and transfer to the real world. DyWA[31], with joint impedance in the action space, also uses RL to train the teacher policy. However, as mentioned before, RL techniques require extra training and configuration, while our method can be directly applied to a human-instantiated demonstration. Frameworks such as [32] also leverage pointcloud observations and contact reward for sim2real dexterous manipulation, but do not explicitly handle continuous contact regulation or adaptive compliance throughout a task.

In contrast to all aforementioned methods, our framework **is the first to show** that one can generate force-informed data with a lightweight trajectory modification approach, such as Laplacian Editing [12] and DS force modulation [10], from a single demonstration in simulation. Such data allows training flow-matching policies on point cloud and force inputs, and thus produces compliant policies that transfer zero-shot to real hardware in contact-rich manipulation tasks.

III. APPROACH

We assume a simulation environment (IsaacGym) is instantiated for each target task, following similar data generation or sim2real setups [26], [27], [33], enabling privileged access to information such as object pose and contact forces. The formulation of our framework starts from

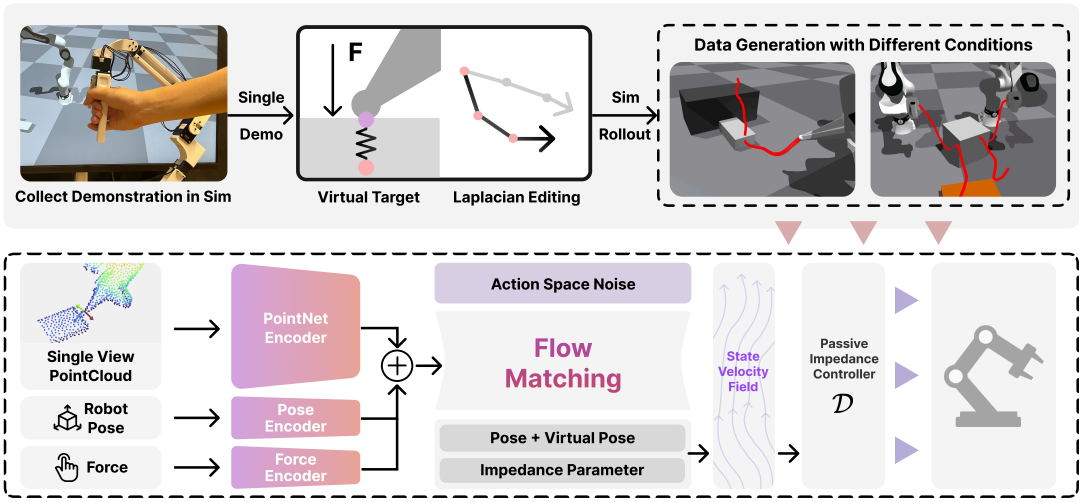


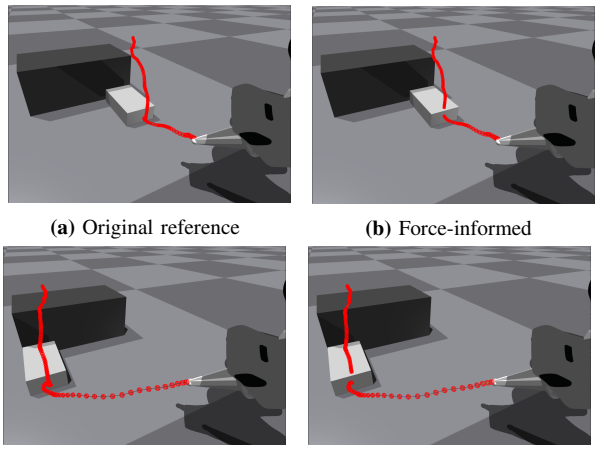
Fig. 2. **Flow with the Force Field** 3D Compliant Visuomotor Policy Learning Framework: Starting from a single simulation demonstration, we augment data by adding force-informed virtual targets and applying Laplacian editing to generate point-cloud and force trajectories beyond the original demo. We train a flow-matching policy that takes point cloud and force as input and predicts actions, including an impedance parameter. At rollout, the policy is synthesized into a state-velocity field and executed with a Passive Impedance Controller for compliant behavior. While our data is only generated with one simple box geometry for both tasks, the trained policies produce generalizable capabilities beyond the single shapes using our framework.

a single example of human demonstration in simulation $D_h = \left(o_h^{(i)}, a_h^{(i)}\right)_{i=1}^{T}$ of length T. The observations $o_h^{(i)} = (x_{ee}^r, F^r, x_{obj})$ are composed of robot end-effector pose $x_{ee}^r \in \mathbb{R}^3 \times \mathcal{SO}(3)$ and end-effector force measurements $F^r \in \mathbb{R}^3$, as well as object pose $x_{obj} \in \mathbb{R}^3 \times \mathcal{SO}(3)$. We define action $a^{(i)}$ as the next end effector pose. We use the privileged information in simulation to help with generating new data. As will be described in Section III-A, D_h is then modified with Laplacian editing and force modulation under domain randomization, which we group the operation as $\mathcal{M}(D_h)$. This operation generates a more diverse and varied simulation dataset of the task $D_{sim} = \left\{\left(o_s^{(i)}, a^{(i)}\right)\right\}_{i=1}^{n}$.

Note that the $o_s^{(i)}$ in D_{sim} is different from the $o_h^{(i)}$ in D_h. The human demonstration in simulation D_h does not collect point cloud information, while D_{sim} does. D_{sim} is used for training the visuomotor compliant flow matching policy $\pi : \mathcal{O} \rightarrow \mathcal{A}$. The observation space for the policy is $\mathcal{O} = (o_{pc}, o_{ee}, o_f)$ and the action space for the policy is $\mathcal{A} = (a_{ee}, d)$, corresponding to the target end-effector pose and compliance gain d. During policy rollout on hardware, to improve safety, robustness, and passivity, especially under sim-to-real discrepancies, we convert the policy output pose $\mathcal{A} = (a_{ee}, d)$ into a state velocity field $\dot{x}_d = f(o_{ee}, d)$. The state velocity field is then fed into a passive impedance controller to produce the desired task space force, which drives the robot, denoted as $F_{ee,d} = D(x)\dot{x}_d$, which damps the deviations from the desired velocity, smoothing motion and suppressing abrupt force spikes; described in Section III-C. Figure 2 shows a complete view of our pipeline.

A. Generating Force-Informed Data from Simulation

We provide a light-weight data generation method for force-sensitive tasks without doing any training but directly warping a single human demonstration under new initial conditions in simulation. As shown in Figure 2, We first use FACTR [9] with force feedback to teleoperate a Franka in IsaacGym and collect a human demonstration. We then split D_h into free-space and in-contact segments (Fig. 3): D_f (length T_f) where $\|F^r\| = 0$, and D_c (length T_c) where $\|F^r\| > 0$. Prior to warping the demonstration trajectory, we apply domain randomization to the object's pose, mass and friction in simulation, yielding new initial conditions $x_{ee}^r(0)$ and $x_{obj}^r(0)$ that differ from those in D_h. Our goal is to generate a new demonstration consistent with these randomized conditions by decomposing and warping D_h.



(a) Original reference **(b)** Force-informed

(c) Laplacian editing **(d)** Force-Laplacian

Fig. 3. Example of trajectory warping with different modulation strategies. (a) is the demonstration trajectory, (b) shows the force-informed trajectory with virtual target, and (c) shows the trajectory warped by object and end effector initial pose, while (d) shows the complete trajectory we use in data generation warped both by object and end effector initial pose and force.

Free-space trajectory: We warp the transition segment of the end-effector trajectory $\mathbf{X}_{ee}^{rf}$ in D_f using Laplacian Editing (LE). LE allows directly modifying an existing trajectory defined by m constrain waypoints $\mathbf{r} \in \mathbb{R}^{d \times m}$ while capturing local properties. Let the free-space positions be $p_{ee}(t)$, $t = 0, \ldots, T_f$. We anchor only two short bands: the

first n_{cs} samples at the start and the last n_{ce} samples at the end ($n_{cs} + n_{ce} = m$). The constraint simply preserves the *relative* offsets of these samples to the new anchors: p_{ee}^{new} at the start and p_{obj}^{new} at the end:

$$\begin{aligned} p_{ee,\mathrm{mod}}^{rf}(t) &= p_{ee}^{new} + \big(p_{ee}(t) - p_{ee}(0)\big), \;\; t = 0{:}n_{cs} - 1, \\ p_{ee,\mathrm{mod}}^{rf}(t) &= p_{obj}^{new} + \big(p_{ee}(t) - p_{ee}(T_f)\big), t = T_f - n_{ce}{:}T_f. \end{aligned} \tag{1}$$

That is, the first few points keep the same displacement from the new initial end-effector position, and the last few points keep the same displacement from the new object position. Then we first convert the constraint waypoints $\mathbf{r}$ in Cartesian space into Laplacian coordinates Δ with the graph Laplacian matrix $L \in \mathbb{R}^{m \times m}$[12],

$$L_{ij} = \begin{cases} 1, & i = j, \\ -\dfrac{w_{ij}}{\sum_{j \in \mathcal{N}_i} w_{ij}}, & j \in \mathcal{N}_i, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

where $\mathcal{N}_i$ are a set of neighbor points $\mathbf{r}_j$ for waypoint $\mathbf{r}_i$, and w_{ij} is a weight set to 1 for this work. One can obtain $\Delta = L\mathbf{r}$, where Δ is a concatenation of the Laplacian coordinate for each waypoint

$$\boldsymbol{\delta}_i = \sum_{j \in \mathcal{N}_i} \frac{w_{ij}}{\sum_{j \in \mathcal{N}_i} w_{ij}} \big(\mathbf{r}_i - \mathbf{r}_j\big). \tag{3}$$

The matrix L can be singular, so one can impose constraints on the system $L\mathbf{r} = \Delta$ when solving for new waypoints $\mathbf{r}$ to achieve editing [12]. To warp the rotational components of the free-space trajectory, we apply SLERP (Spherical Linear Interpolation of Rotations) between q_{ee}^{new} and q_{obj}^{new}, interpolating the rotation part to the same length of T_f which yields a smooth rotational transition over the entire free-space segment. Combining these orientations with the warped positions gives the complete modulated trajectory $\mathbf{X}_{ee_{mod}}^{rf} \in \mathbb{R}^3 \times \mathcal{SO}(3)$. As illustrated in Fig. 3(a)–(c), the object-pose modulation stretches the free-space path and smoothly reorients it in accordance with the object's pose.

In-contact trajectory: For the in-contact part, we maintain an object-centric frame for trajectory warping. Let $\mathbf{X}_{ee}^{rc}, \mathbf{X}_{obj}^{c}$ denote the in-contact end-effector and object pose trajectory in the demonstration, and $x_{obj}^{new} = [p_{obj}^{new}, q_{obj}^{new}]$ the randomized initial object pose in simulation. For every data point, the warped end-effector pose $\mathbf{X}_{ee_w}^{rc}(i) = [\mathbf{P}_{ee_w}^{rc}(i), \mathbf{Q}_{ee_w}^{rc}(i)]$ and force $F_{warp}^{r}(i)$ are given by:

$$\mathbf{P}_{ee_w}^{rc}(i) = p_{obj}^{new} + \mathbf{Q}_{obj}^{c}(i)^{-1}(\mathbf{P}_{ee}^{rc}(i) - \mathbf{P}_{obj}^{c}(i)), \tag{4}$$

$$\mathbf{Q}_{ee_w}^{rc}(i) = q_{obj}^{new}\, \mathbf{Q}_{obj}^{c}(i)^{-1}\, \mathbf{Q}_{ee}^{rc}(i), \tag{5}$$

$$F_{warp}^{r}(i) = q_{obj}^{new}(i)\, \mathbf{Q}_{obj}^{c}(i)^{-1}\, F^{r}(i), \tag{6}$$

where $\mathbf{P}_{obj}^{c}$ and $\mathbf{Q}_{obj}^{c}$ are the positions and quaternions from the in-contact part object trajectory $\mathbf{X}_{obj}^{c}$. Equation (4)(5) preserves the relative end-effector pose in the object frame while Equation (6) rotates the demonstrated force into the randomized object frame. Inspired by [10], [11], which use an end effector velocity orthogonal to the contact surface to do force modulation, and [4], [5], which steer toward a virtual target in the measured force direction, we construct a force-informed virtual target x_{ee}^{v} for each data point $\mathbf{x}_{ee_w}^{rc}$ in $\mathbf{X}_{ee_w}^{rc}$ as:

$$x_{ee}^{v} = x_{ee_w}^{rc} + k_f F^{r} \tag{7}$$

where k_f is a hand-tuned parameter controlling the modulation strength. We claim that without the force-informed virtual target tracking, the task is not able to succeed because it has no intention of maintaining continuous contact. An analogy for the virtual target is a target pose inside the object. Moving to that target will generate force on the object surface. Comparing Figure 14(a) and 14(b), we could see that the force-informed trajectory gets into the object surface after adding the virtual target. In our method, the compliance scheduling term d is supervised by force feedback: during policy learning, the contact force measured in simulation is converted to a damping target through a fixed linear mapping, and the flow-matching policy is trained to predict d to match this target, which will be introduced in Section III-C.

B. Compliant 3D Visuomotor Flow Matching Policy

To learn visuomotor policies from simulation-generated data, we use a flow matching framework [17], [16], [18] as our core method. Flow matching runs at a higher frequency than diffusion-based models, making it suitable for contact-rich tasks. Prior work has shown that pointcloud inputs significantly enhance data efficiency and spatial generalization [15], [29]. Accordingly, we adopt a single-view point cloud as input, eliminating reliance on privileged object pose information. Thus, our observation space is defined as $\mathcal{O} = (o_{pc}, o_{ee}, o_f)$.

We use DP3 Encoder from [15] to encode the point cloud o_{pc}. We also follow the same practice for processing the data, including furthest point sampling to 1024 points as well as cropping for the workspace. To be more contact-aware, we also include force o_f as part of the observation, which we use a simple 3-layer MLP to output a 64-D feature vector. The input also includes a feature vector encoded by a 3-layer MLP for the robot pose o_{ee}. We then concatenate the three feature vectors as the observation condition.

The flow matching backbone in Fig. 2, a conditional U-Net same as [15], but with a different output space. In order to achieve compliant motion with contact in the environment, we augment the flow matching action space with a virtual pose trajectory $X_{vt}^{t\ldots t+H}$ and an impedance parameter trajectory $d^{t\ldots t+H}$. Therefore, the policy output is $A = (X_{ref}^{t\ldots t+H}, X_{vt}^{t\ldots t+H}, d^{t\ldots t+H})$, which are the robot end effector reference pose trajectory $X_{ref}^{t\ldots t+H}$, virtual pose trajectory $X_{vt}^{t\ldots t+H}$, as well as impedance parameter trajectory $d^{t\ldots t+H}$. We use $H = 16$ as the policy output horizon in this work. We will describe the reference pose, virtual pose, and the impedance parameter in detail in Section III-C. Next, we formally define the flow matching component.

The goal of conditional flow matching is to estimate a vector field $\boldsymbol{v_\theta} : \mathbb{R}^d \times [0, 1] \to \mathbb{R}^d$, such that integrating the Ordinary Differential Equation (ODE)

$$\frac{\mathrm{d}z_t}{\mathrm{d}t} = \boldsymbol{v_\theta}\left(z_t, t\right), \tag{8}$$

on time $t \in [0,1]$ transports z_0 from base distribution p_0 to the target distribution z_1 from p_1. The base distribution p_0 is typically chosen as Gaussian noise $\mathcal{N}(0, I)$. To learn such a flow, training proceeds by matching $\boldsymbol{v_\theta}$ along an interpolation path $z_t = tz_1 + (1-t)z_0$. By taking the derivative, one will obtain the direction of a linear path $\boldsymbol{u} = z_1 - z_0$, which is the target velocity. We parametrize the $\boldsymbol{v_\theta}$ with a conditional U-Net and matching the $\boldsymbol{u}$ and $\boldsymbol{v_\theta}$ by solving the following regression problem [16],

$$\min_{\boldsymbol{v}} \mathbb{E}_{z_0 \sim p_0, z_1 \sim p_1} \left[\int_0^1 \left\| \boldsymbol{u}\left(z_1, z_0\right) - \boldsymbol{v}_\theta\left(z_t, t\right) \right\|_2^2 \, \mathrm{d}t \right]. \quad (9)$$

In this work, the target sample z_1 is the action in trajectory data D_{sim} generated from the simulation. During inference, we first initiate a noisy action sample A_0 from p_0. Then the ODE is integrated with Euler from $t = 0$ to $t = 1$ conditioned on the observation $\mathcal{O}$ includes pointcloud o_{pc}, end-effector pose o_{ee}, and end-effector force o_f,

$$A_{t+\delta} = A_t + \delta \boldsymbol{v_\theta}(A_t, \mathcal{O}), \quad (10)$$

which we set the $\delta = 0.1$ for all of our experiments. After obtaining the actions rollout $A = (X_{ref}^{t\ldots t+H}, X_{vt}^{t\ldots t+H}, d^{t\ldots t+H})$, rather than tracking this trajectory with classical position-based impedance controller as commonly done in prior works, we introduce a different compliant rollout technique in the next section.

C. Compliant Policy Rollout with State Vector Field

After obtaining the policy rollout, we execute the resulting trajectory using a passive impedance controller in task space [19]. This controller ensures passive interaction with the environment, which is crucial when contact leads to constraints or sticking. Although the impedance parameter has been referenced as part of the policy action, we now formalize it. We begin with the control law from [19]:

$$F_c = G_x(x) - D(x)(\dot{x} - f(x)), \quad (11)$$

where $G_x(x)$ is the gravity compensation term (omitted from now on), $D(x) \in \mathbb{R}^{d \times d}$ is the damping matrix, $\dot{x} \in \mathbb{R}^d$ is the current end-effector velocity, and $f(x) = \dot{x}_d$ is the desired velocity from the policy. The damping matrix $D(x)$ is structured via an eigendecomposition: $D(x) = V(x)\Lambda V(x)^\top$, where Λ is a diagonal matrix of damping gains and $V(x) = [v_1, v_2, v_3]$ is an orthonormal matrix of eigenvectors, with v_1 aligned with the desired direction $f(x)$.

While damping-only impedance has the advantage of passivity, reducing damping sacrifices trajectory tracking, potentially causing distribution shifts. Moreover, tuning directional damping gains often requires significant manual effort and may degrade performance. To avoid these, we fix the damping magnitude and instead modulate the desired velocity direction to shape compliance. Unlike the flow matching vector field, this vector field is in the end effector task space during the execution stage. Specifically, we consider only the desired velocity component from Equation 11:

$$F_d = D(x)f(x). \quad (12)$$

where $f(x)$ is collinear with v_1, which we can simplify to $F_d = v_1 f(x)$. We then decompose the desired velocity into magnitude and direction: $f(x) = k\hat{u}$, where $k = 0.1$ is the desired velocity magnitude, which can be adjusted online or designed, and $\hat{u} \in \mathbb{R}^d$ is the direction vector, which is your current desired moving direction.

Let $x_{curr} \in R^N$ be the robot's current end effector position, the rollout reference trajectory from flow matching be $X_{ref}^{t\ldots t+H}$, and the virtual target trajectory be $X_{vt}^{t\ldots t+H}$. Each reference X_{ref}^t has one corresponding virtual target X_{vt}^t, which $X_{vt}^t - X_{ref}^t$ provides the compliant direction at X_{ref}^t. Therefore, we now define $\hat{t} = X_{ref}^{t+1} - X_{ref}^t$ and the compliant direction $\hat{n} = X_{vt}^t - x_{curr}$. In ideal case, $x_{curr} \approx X_{ref}^t$. We construct $\hat{u}$ to blend between the reference motion direction $\hat{t}$ and compliant direction $\hat{n}$:

$$\hat{u} = \frac{d\hat{n} + \hat{t}}{||d\hat{n} + \hat{t}||}, \quad (13)$$

where $d \in \mathbb{R}_{>0}$ is a gain controlling convergence toward the virtual target and the degree of compliance. This gain is the impedance parameter of our flow-matching policy. During training, the gain d is scheduled based on the estimated force magnitude $|\mathcal{F}|$ collected by the force sensor in the simulation. Similar to the stiffness in [4]. We define it as:

$$d = \begin{cases} d_\uparrow, & |\mathcal{F}| < \mathcal{F}\downarrow \\ d\uparrow - (d_\uparrow - d_\downarrow)\frac{|\mathcal{F}| - \mathcal{F}\downarrow}{\mathcal{F}\uparrow - \mathcal{F}\downarrow}, & \mathcal{F}\downarrow \le |\mathcal{F}| \le \mathcal{F}\uparrow \\ d\downarrow, & |\mathcal{F}| > \mathcal{F}\uparrow \end{cases} \quad (14)$$

where $\uparrow$ and $\downarrow$ are hardware-specific maximum and minimum values. This formulation enables directional compliance to emerge naturally from modulating $f(x)$, without sacrificing tracking fidelity. While this formulation might break passivity with such usage, we argue that this state-dependent vector field rollout will allow safer interruption during execution when interacting with the environment. Furthermore, we will show in the experiments that this formulation actually offers better performance in the real world contact tasks while at the same time injecting less energy.

IV. EXPERIMENT

Our experiments aim to answer the following questions:

- What data can our force-informed data generation scheme offer for learning visuo-force policies?
- How well can the policy adapt to the real world with generated data (in terms of spatial and object generalization and matching the desired profile)?
- How does the passive impedance controller perform compared to the classical impedance controller for rolling out the adaptive compliant visuo-force policy?

We evaluate our pipeline on two real-world tasks: **Block Flipping** and **Bi-manual Moving**. In **Block Flipping**, a single arm pivots a block from flat to upright ($\approx 85°$) and must keep it stable. In **Bi-manual Moving**, two arms cooperatively place a large object onto a goal platform; a trial succeeds when the object is stably stacked. Up to three self-recoveries are allowed per trial.

TABLE I

DOMAIN RANDOMIZATION TESTING RANGE

	Block Flipping	**Bi-manual Moving**
Robot End Effector [cm]	[±15, ±5, ±3]	[±15, ±5, 0.0]
Object Translation [cm]	[±15, ±5, 0.0]	[±10, ±15, 0.0]
Object Orientation [°]	[0.0, 0.0, 0.0]	[0.0, 0.0, ±20]
Object Mass [kg]	[0.1, 0.8]	[0.1, 0.8]
Friction	[0.2, 1.0]	[0.2, 1.0]
Success Rate [%]	97.6	82.9

Data are collected in IsaacGym using FACTR [9] for teleoperation; real-world runs use Franka arms. End-effector force is inferred from Franka joint torques, and perception uses an Intel RealSense L515 LiDAR. To reduce the Sim2Real gap for the point cloud data, we inject noise into simulated point clouds, such as vertical "flying pixels" and occlusion shadows by detecting depth jumps and interpolating across them. We also normalize simulated force, which the real forces are also normalized during real deployment.

A. Generating Force-Informed and Point Cloud Data

To evaluate our simulation-based data generation method, we conduct experiments on **Block Flipping** and **Bi-manual Moving** with domain randomization over object pose, mass, friction, and end-effector pose (Table I). We intentionally omit object shape variations to test whether simple simulation geometry, combined with our proposed components, suffices for robust real-world deployment.

For **Block Flipping**, 303 trajectories are generated for policy learning. During the data generation, the robot follows the reference trajectory as shown in (Fig. 2) , switches to object-centric replay upon contact, and tracks force-informed virtual targets with a forward speed to maintain contact and complete the flipping stroke. This achieves a success rate of 97.6%. For **Bi-manual Moving**, 210 trajectories are generated. Both arms follow the same pipeline, using object-centric replay and force-informed targets to maintain coordinated grasps. Quaternion interpolation with SLERP ensures smooth reorientation even under box translation and rotation, yielding robust $SE(3)$-adaptive behavior with 82.9% success. In both tasks, removing Laplacian editing or force-informed targets reduces the success rate to 0%.

B. Deploying the Compliant Policy in the Real World

Given the generated simulation data, the policy is trained with 3000 epochs. The inference machine uses an RTX 3090 Ti GPU. We evaluated the trained policy in real-world settings to demonstrate spatial and object generalization. We compare to ablation baselines to show that our design is superior for such tasks. While works like [4] and [5] have demonstrated the importance of force-informed virtual target, their tasks are with RGB data, naturally lacking 3D understanding. Since our work utilizes point clouds, we ask: would force-informed data still be necessary, given that we have access to the geometric features? How important is force input in these tasks? To answer these questions we provide three baselines: (1) **3D FM**: A point cloud-only flow matching adapted directly from [15], (2) **3D FM Comp**: A point cloud-only flow matching with compliant output, (3) **3D FM Force**: Using point cloud and force as input without compliant output, and (4) **3D FM Comp + Force (*Ours*)**: Our full approach. Success rates reported in Table II.

TABLE II

SUCCESS RATES FOR SPATIAL AND OBJECT GENERALIZATION

Method	**Block Flipping**		**Bi-Manual Moving**	
	Spatial	Object	Spatial	Object
3D FM	0/27	0/12	6/10	3/9
3D FM Comp	0/27	0/12	**8/10**	**7/9**
3D FM Force	16/27	4/12	7/10	6/9
3D FM Comp + Force	**24/27**	**10/12**	**8/10**	**7/9**

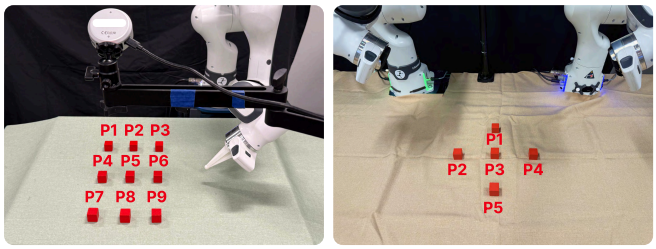
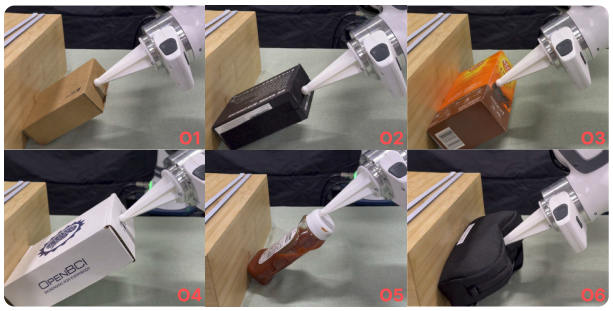


Fig. 4. The red cubes show the location for spatial performance evaluation on (left) **Block Flipping** and (right) **Bi-manual Moving**.

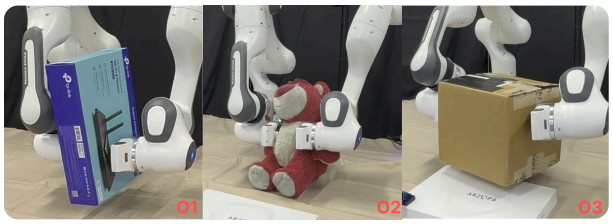
1) Block Flipping: We evaluate real-world spatial generalization at nine testing positions (Fig. 4), as illustrated in Fig. 4. Each numbered position represents a testing site where the center of the test object (Object O1, shown in Fig. 5(a)) is placed. At each location, we run three trials per method (totaling 27 trials). **3D FM** and **3D FM Comp** had a 0% success rate, primarily failing due to difficulties in establishing initial contact or getting stuck due to excessive force. **3D FM Force** succeeded consistently in central positions (P4, P5, P6). However, its performance notably degrades on the sides, getting stuck during flipping due to rigid interactions. Our method achieved consistent success at all locations except P7, likely due to limited training coverage or kinematic constraints. This evaluation shows that our method can generalize spatially in the real world by using one demonstration in the simulation.

One might question that using a simple box geometry in generating point cloud data won't generalize across objects in the real world. We use only one size box geometry to generate data and train the policy. We then test it in the real world across objects as shown in Fig. 5(a). We tested all objects at P5 with two different seeds. The last column of Table II shows the success rate. Again, **3D FM** and **3D FM Comp** fail for all trials. The failure modes are similar to the spatial evaluation. **3D FM Force** succeeds at O1 and O6 twice, but fails at others, either not making contact (O3, O5) or getting stuck (O2, O4). Our method fails at O5 once, which cannot make contact, and O3 once, which slips back. This shows that even training with simple geometry, point cloud and force could be combined with the adaptive compliant output to allow for better execution.

Fig. IV-B.2 shows the force and compliance profiles comparing simulation data and real-world rollout on one example run. The real-world force profiles are scaled to match simulation magnitude for visualization purposes. The



(a) Objects for *Block Flipping* in Real World



(b) Objects for *Bi-manual Moving* in Real World

Fig. 5. Real-world object and spatial generalization results.

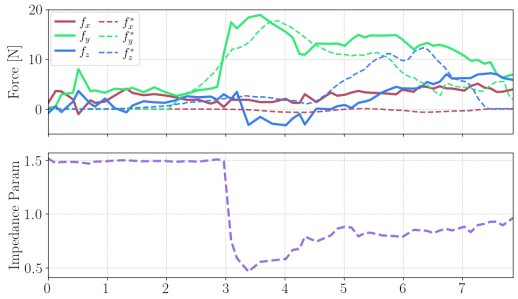


Fig. 6. Force and impedance profiles from **Block Flipping** show similar trends between simulation and real. The solid lines are the scaled and smoothed real measurements, while the dashed lines are the force from the simulation demonstration.

TABLE III

COMPLIANT CONTROLLER COMPARISON

Controller Type	**Success Rate**	**Energy Mean (J)**
Classical Impedance	8/12	0.835
Passive Impedance	**10/12**	**0.734**

plots show that the learned policy deployed in the real world can match the desired behavior in simulation. The f_y increases upon making the first contact and then decreases, while f_z starts to increase. The bottom plot shows the change of the impedance parameter d over time. Upon making the first contact, d decreases to become more compliant and then slowly increases afterwards, matching our design.

2) Bi-manual Moving: We first evaluate real-world spatial generalization at five distinct locations, as shown in Fig. 4, using the O3 object in Fig. 5(b), with each location having two runs. At each run, the object orientation is perturbed slightly to provide more variations. The success rate is shown in Table II. The baseline **3D FM** performs well in the spatial task, with failures happening at P1 and P2. At P1, the object is too close, and both arms tend to move towards the center and quickly forward without the idea of touching the object. Also, **3D FM** does not perform well on objects O1 and O2, whose sizes are smaller than the training point cloud. On the other hand, **3D FM Comp** performs well. The outputting virtual target provides a closer grasping distance between the two arms, creating a bigger chance of locking the object. The failures are commonly on the two sides (P2 and P4) with object O3, where the robot cannot avoid the edge of the box when coming down from above to grasp the side. They tend to get stuck on top of the box. Our method and **3D FM Force** also have the same failure source. The learned policy exhibits recovering behavior from failures. Even when the object drops before the goal platform, the policy recovers by grasping the object again and moving forward until it is on the platform. Such an emergent recovering behavior makes the policy robust in the real world. The goal platform is at a location where the robots can easily get close to singularity in order to place the object on it, so such a readjustment after failure improves the success rate a lot. For different objects, we perform tests with three runs for each object at P5. All methods have a failure with object O1. This object is difficult to recover from and can easily lose control due to its height. **3D FM Force** fails once with O3, which moves forward without touching the box to obtain the force signal. Force and impedance profiles for this task are in the video.

C. Passive Impedance for Real World

We evaluate the effectiveness of passive impedance control in deploying visuomotor policies trained exclusively in simulation. Due to discrepancies and noise inherent in sim2real transfer, policies often exhibit inaccuracies when executed in the real world. We conducted a comparative study between a classical position-based adaptive impedance controller and our proposed passive impedance controller. We tested both controllers using the same trained checkpoint across six distinct objects in a real-world block-flipping task. Each object was consistently placed at P5, and each controller was executed twice per object, resulting in 12 trials.

Our results indicate that passive impedance control achieves a notably higher success rate compared to the classical impedance approach. Specifically, the classical controller encountered two failures with O4 in Fig. 5(a), which is an object significantly larger than those encountered during training. For these OOD cases, the policy's reference trajectory penetrated the object's geometry. The classical controller, tracking position setpoints, first pushes it hard to attempt to achieve these unreachable positions, then starts to lower the stiffness and progresses to the lift-up stage. In contrast, the passive impedance controller, leveraging velocity-based control, naturally yielded without enforcing strict positional constraints, successfully adapting and completing the task. Additionally, we measured the total energy injected into the environment during successful trials in the period of making contact. We show that passive impedance control injects less energy on average over different objects, highlighting its compliance during real-world execution.

V. DISCUSSION & CONCLUSION

We have presented a full framework for (1) generating force and demonstration-informed simulation data, (2) learning 3D adaptive compliant flow matching policies, and (3) rolling out compliant policies with a passive impedance controller using a state vector field. Our experiments show that the policy can adjust its compliance in response to both point cloud and force inputs, enabling more robust execution in tasks with continuous contact than methods that rely solely on positional accuracy. Zero-shot deployment on real Franka robots demonstrates that even without real-world data, the policy executes reliably in both tasks. We also find that the passive compliance rollout reduces energy injection and even improves success. Some limitations remain for future work. For example, creating a simulation environment could require some effort for complicated tasks. Also, our work does not adapt to objects with very different physical properties. Last but not least, our method considers the point cloud only, which cannot achieve tasks that require color information. Overall, our results suggest that a simple sim-based data generation scheme paired with force-aware, compliant execution can significantly reduce the dependency on real-world data while maintaining performance.

REFERENCES

[1] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *The International Journal of Robotics Research*, p. 02783649241273668, 2023.

[2] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," *arXiv preprint arXiv:2304.13705*, 2023.

[3] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter *et al.*, "π_0: A vision-language-action flow model for general robot control," *arXiv preprint arXiv:2410.24164*, 2024.

[4] Y. Hou, Z. Liu, C. Chi, E. Cousineau, N. Kuppuswamy, S. Feng, B. Burchfiel, and S. Song, "Adaptive compliance policy: Learning approximate compliance for diffusion guided control," *arXiv preprint arXiv:2410.09309*, 2024.

[5] C. Chen, Z. Yu, H. Choi, M. Cutkosky, and J. Bohg, "Dexforce: Extracting force-informed actions from kinesthetic demonstrations for dexterous manipulation," *IEEE Robotics and Automation Letters*, 2025.

[6] M. Aburub, C. C. Beltran-Hernandez, T. Kamijo, and M. Hamaya, "Learning diffusion policies from demonstrations for compliant contact-rich manipulation," 2024. [Online]. Available: https://arxiv.org/abs/2410.19235

[7] Y. Wang, C. C. Beltran-Hernandez, W. Wan, and K. Harada, "Hybrid trajectory and force learning of complex assembly tasks: A combined learning framework," *IEEE Access*, vol. 9, pp. 60 175–60 186, 2021.

[8] M. Kim, S. Niekum, and A. Deshpande, "Scape: Learning stiffness control from augmented position control experiences," in *Conference on Robot Learning. Proceedings of Machine Learning Research*, 2021.

[9] J. J. Liu, Y. Li, K. Shaw, T. Tao, R. Salakhutdinov, and D. Pathak, "Factr: Force-attending curriculum training for contact-rich policy learning," *arXiv preprint arXiv:2502.17432*, 2025.

[10] W. Amanhoud, M. Khoramshahi, and A. Billard, "A dynamical system approach to motion and force generation in contact tasks." in *Robotics: Science and Systems*, vol. 2019, 2019.

[11] W. Amanhoud, M. Khoramshahi, M. Bonnesoeur, and A. Billard, "Force adaptation in contact tasks with dynamical systems," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 6841–6847.

[12] T. Nierhoff, S. Hirche, and Y. Nakamura, "Spatial adaption of robot trajectories based on laplacian trajectory editing," *Autonomous Robots*, vol. 40, no. 1, pp. 159–173, 2016.

[13] T. Li and N. Figueroa, "Task generalization with stability guarantees via elastic dynamical system motion policies," in *7th Annual Conference on Robot Learning*, 2023.

[14] T. Li, S. Sun, S. S. Aditya, and N. Figueroa, "Elastic motion policy: An adaptive dynamical system for robust and efficient one-shot imitation learning," *arXiv preprint arXiv:2503.08029*, 2025.

[15] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu, "3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations," in *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

[16] E. Chisari, N. Heppert, M. Argus, T. Welschehold, T. Brox, and A. Valada, "Learning robotic manipulation policies from point clouds with conditional flow matching," *arXiv preprint arXiv:2409.07343*, 2024.

[17] X. Liu, C. Gong *et al.*, "Flow straight and fast: Learning to generate and transfer data with rectified flow," in *The Eleventh International Conference on Learning Representations*.

[18] Q. Zhang, Z. Liu, H. Fan, G. Liu, B. Zeng, and S. Liu, "Flowpolicy: Enabling fast and robust 3d flow-based policy via consistency flow matching for robot manipulation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 14, 2025.

[19] K. Kronander and A. Billard, "Passive interaction control with dynamical systems," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 106–113, 2015.

[20] A. Billard, S. S. Mirrazavi Salehian, and N. Figueroa, *Learning for Adaptive and Reactive Robot Control: A Dynamical Systems Approach*. Cambridge, USA: MIT Press, 2022.

[21] Z. He, H. Fang, J. Chen, H.-S. Fang, and C. Lu, "Foar: Force-aware reactive policy for contact-rich robotic manipulation," *IEEE Robotics and Automation Letters*, 2025.

[22] H. Xue, J. Ren, W. Chen, G. Zhang, Y. Fang, G. Gu, H. Xu, and C. Lu, "Reactive diffusion policy: Slow-fast visual-tactile policy learning for contact-rich manipulation," *arXiv preprint arXiv:2503.02881*, 2025.

[23] W. Liu, J. Wang, Y. Wang, W. Wang, and C. Lu, "Forcemimic: Force-centric imitation learning with force-motion capture system for contact-rich manipulation," *arXiv preprint arXiv:2410.07554*, 2024.

[24] 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," in *2019 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2019, pp. 1010–1017.

[25] X. Zhang, L. Sun, Z. Kuang, and M. Tomizuka, "Learning variable impedance control via inverse reinforcement learning for force-related tasks," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2225–2232, 2021.

[26] L. Yang, H. T. Suh, T. Zhao, B. P. Græsdal, T. Kelestemur, J. Wang, T. Pang, and R. Tedrake, "Physics-driven data generation for contact-rich manipulation via trajectory optimization," *arXiv preprint arXiv:2502.20382*, 2025.

[27] J. A. Barreiros, A. Ö. Önol, M. Zhang, S. Creasey, A. Goncalves, A. Beaulieu, A. Bhat, K. M. Tsui, and A. Alspach, "Learning contact-rich whole-body manipulation with example-guided reinforcement learning," *Science Robotics*, vol. 10, no. 105, p. eads6790, 2025.

[28] S. Peri, I. Lee, C. Kim, L. Fuxin, T. Hermans, and S. Lee, "Point cloud models improve visual robustness in robotic learners," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 17 529–17 536.

[29] Z. Xue, S. Deng, Z. Chen, Y. Wang, Z. Yuan, and H. Xu, "Demogen: Synthetic demonstration generation for data-efficient visuomotor policy learning," *arXiv preprint arXiv:2502.16932*, 2025.

[30] M. Noseworthy, B. Tang, B. Wen, A. Handa, C. Kessens, N. Roy, D. Fox, F. Ramos, Y. Narang, and I. Akinola, "Forge: Force-guided exploration for robust contact-rich manipulation under uncertainty," *IEEE Robotics and Automation Letters*, 2025.

[31] J. Lyu, Z. Li, X. Shi, C. Xu, Y. Wang, and H. Wang, "Dywa: Dynamics-adaptive world action model for generalizable non-prehensile manipulation," *arXiv preprint arXiv:2503.16806*, 2025.

[32] Y. Qin, B. Huang, Z.-H. Yin, H. Su, and X. Wang, "Dexpoint: Generalizable point cloud reinforcement learning for sim-to-real dexterous manipulation," *Conference on Robot Learning (CoRL)*, 2022.

[33] T. G. W. Lum, M. Matak, V. Makoviychuk, A. Handa, A. Allshire, T. Hermans, N. D. Ratliff, and K. Van Wyk, "Dextrah-g: Pixels-to-action dexterous arm-hand grasping with geometric fabrics," in *Conference on Robot Learning*. PMLR, 2025, pp. 3182–3211.