
White Rose Research Online

eprints@whiterose.ac.uk
https://eprints.whiterose.ac.uk
Universities of Leeds, Sheffield and York

Deposited via The University of Leeds.

White Rose Research Online URL for this paper:
https://eprints.whiterose.ac.uk/id/eprint/247118/

Version: Accepted Version

Article:
Wu, S., Liu, C., Wang, Y. et al. (2026) Robust Hybrid Diffusion: Unifying Task Space and Joint Space for Robot Manipulation. IEEE Robotics and Automation Letters, 11 (12). 13853 -13860. ISSN: 2377-3766

https://doi.org/10.1109/lra.2026.3745015

This is an author produced version of an article published in IEEE Robotics and Automation Letters, made available via the University of Leeds Research Outputs Policy under the terms of the Creative Commons Attribution License (CC-BY), which permits unrestricted use, distribution and reproduction in any medium, provided the original work is properly cited.

Reuse
This article is distributed under the terms of the Creative Commons Attribution (CC BY) licence. This licence allows you to distribute, remix, tweak, and build upon the work, even commercially, as long as you credit the authors for the original work. More information and the full terms of the licence here: https://creativecommons.org/licenses/

Takedown
If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.





Robust Hybrid Diffusion: Unifying Task Space and Joint Space for Robot Manipulation

Shaokang Wu[1], Chen Liu[1], Yijin Wang[1], Chuankai Zhang[1], Linyan Han[2], and Yanlong Huang[1]

***Abstract*—Diffusion policies have demonstrated strong capability in learning multimodal action distributions in the context of imitation learning. While human demonstrations in general consider both task accuracy and joint configuration simultaneously, existing approaches are typically restricted to either task space or joint space, overlooking the multimodality exhibited in both spaces. To take a pushing task performed by a redundant robot as an example, it is highly desirable for the robot to push an object accurately toward its target while preserving diverse and demonstration-consistent joint-motion patterns. In addition, diffusion policies usually execute action sequences in an open-loop manner, leaving them susceptible to external disturbances (e.g., the change of payload or human intervention). To address these challenges, we propose *robust hybrid diffusion* (RhD), a novel imitation learning framework that unifies diffusion policies in both Cartesian space and joint space, where dynamical movement primitives are leveraged to ensure the robustness of robot actions. Evaluations on three simulated tasks and a real-world pushing task show that RhD achieves high success rates (even under external perturbations) with multimodal end-effector poses and joint configurations.**

***Index Terms*—Imitation learning, redundant robots, diffusion policies, dynamical movement primitives**

I. Introduction

DIFFUSION policies [1], [2], which formulate action generation as a conditional denoising process (e.g., conditioned on image observations and language instructions), have demonstrated the ability to model multimodal actions, scale to high-dimensional action space, and maintain strong training stability. These advantages are crucial when applying imitation learning to transfer human skills to robots.

Although remarkable success with diffusion policies has been reported, this line of work is usually designed to learn either end-effector poses or joint angles as actions [1], [3], [4]. Indeed, action learning in a single space (Cartesian space or joint space) is sufficient for many tasks; however, such a treatment may become inappropriate for tasks that require the learning of both spaces simultaneously. To take the peg-in-hole task as an example, generating end-effector actions via diffusion models is more likely to yield a higher task success rate than generating joint-space actions [5], because successful insertion requires precise alignment of the end-effector's pose with the hole, and such a requirement is represented explicitly in Cartesian space rather than joint space. However, a task-space policy leaves the redundant joint configurations unspecified and may therefore fail to preserve the diverse joint-motion patterns present in the demonstrations. In contrast, training diffusion policies to produce joint actions can preserve multimodal joint configurations, but may come at the cost of a lower success rate [6].

Furthermore, previous works on diffusion policies usually generate and execute a short sequence (i.e., a chunk) of actions without intermediate replanning [1], [3], [7], rather than predicting a single-step action at each time step [8], [9]. While this treatment promotes the temporal consistency of actions, it has a limited capacity for unexpected disturbances since the short action sequence is essentially executed in an open-loop way. If the robot's proprioceptive state changes dramatically due to sudden human interventions, blindly following the pre-planned chunk can result in excessive accelerations and even lead to task failure. To endow diffusion policies with robustness against external perturbations, it is important to integrate the robot's state into action planning and adjust its actions online whenever needed.

In this paper, we propose *robust hybrid diffusion* (RhD) to cope with the simultaneous learning in both Cartesian space and joint space, as well as the robustness requirement. Specifically, we learn actions in Cartesian space and joint space with two independent diffusion policies. In both spaces, dynamical movement primitives (DMP) [10] are leveraged to ensure the robustness (Section IV-A) since DMP offers a well-built convergence property allowing for handling out-of-distribution states and recovering from disturbances during execution. To unify diffusion policies in both spaces, we exploit the Jacobian-based null space, enabling high task success rate and maintaining multimodal end-effector poses and joint configurations (Section IV-B). We verify the effectiveness of RhD through several simulation tasks and a real-world pushing task, where a series of baselines, including those that consider only a single space, are studied (Section V).

II. Related Work

A. Diffusion Policies for Imitation Learning

Diffusion policies have been widely adopted in imitation learning [2], [3]. In [11], hierarchical diffusion policies were introduced, where a high-level policy predicted contact points and a low-level policy was responsible for generating action trajectories. In [12], a keyframe-based diffusion framework

Manuscript received: May 6, 2026; Revised September 3, 2026; Accepted October 7, 2026. This paper was recommended for publication by Editor Lorenzo Jamone upon evaluation of the Associate Editor and Reviewers' comments. *(Corresponding authors: Yijin Wang and Yanlong Huang.)*

[1]Shaokang Wu, Chen Liu, Yijin Wang, Chuankai Zhang, and Yanlong Huang are with the School of Computer Science, University of Leeds, Leeds LS2 9JT, UK (e-mail: y.l.huang@leeds.ac.uk).

[2]Linyan Han is with the School of Mathematics, University of Leeds, Leeds LS2 9JT, UK.

Digital Object Identifier (DOI): see top of this page.

was developed, and macro-actions were employed to construct long-horizon trajectories. Although these approaches improve sample efficiency and task success rates, they primarily focus on task space learning (i.e., generating end-effector actions via diffusion models), neglecting the imitation of joint behaviours presented in demonstrations. While joint-space diffusion policies can capture multimodal joint motions [4], [13], their training objective directly constrains joint actions rather than task-space errors. Therefore, task-specific Cartesian accuracy cannot be guaranteed [6]. To address this challenge, [6] proposed robot kinematics diffuser (RK-Diffuser), where the predicted joint actions were iteratively refined toward minimizing the difference between their corresponding end-effector poses (calculated using forward kinematics) and the predicted end-effector poses. While this approach attains the diversity in joint space and a high success rate, its reliance on iterative, post-hoc optimization imposes high computational demands. In this paper, we leverage *null-space control* to derive an *analytical solution* that unifies end-effector and joint actions, thereby eliminating iterative post-hoc optimization.

B. Robust Policies for Imitation Learning

Prior studies have adopted action chunking in diffusion policies to generate temporally consistent behaviour [1], [7], [14]. However, the resulting policies are susceptible to external perturbations when executing these action chunks in an open-loop manner. To enhance robustness against disturbances or imperfect executions, high-frequency tactile feedback was combined with the outputs of the diffusion model to adjust actions online in [15]. While effective, this approach increases system complexity owing to additional tactile sensors and a separately trained reactive module.

An alternative approach to achieve robustness is to generate actions through an autonomous dynamical system (DS), which can handle out-of-distribution states and resist unpredictable disturbances since the convergence property of DS is theoretically guaranteed. In [16], the attractor's position was predicted from image observations, and later actions converging to the attractor were generated via DMP even under external perturbations. However, this method hinges on the assumption that a unimodal distribution of the attractor, which hinders its applicability in tasks involving multimodal goals. In [3], a variant of DMP [17] was integrated into diffusion models to generate actions in Cartesian space. This method predicted the whole sequence of actions without intermediate replanning during task execution. Although it improves data efficiency and motion smoothness, it essentially corresponds to an open-loop solution and neglects the convergence property offered by DMP. In this paper, we leverage the multimodal feature of diffusion models and the convergence feature of DMP, which enables *real-time correction under perturbations* and preserves the *multimodal actions*.

III. Preliminaries

A. Dynamical Movement Primitives

DMP provides a flexible framework for generating goal-oriented trajectories by coupling a first-order canonical system with a second-order transformation system [10]. Its standard formulation is

$$\begin{aligned} &\tau \dot{s} = -\alpha_x s, \\ &\tau^2 \ddot{\boldsymbol{\xi}} = \alpha_z(\beta_z(\mathbf{g} - \boldsymbol{\xi}) - \tau \dot{\boldsymbol{\xi}}) + s\mathbf{f}_{\mathbf{w}}(s), \\ &f^k_{\mathbf{w}}(s) = \frac{\sum_{h=1}^{H} \varphi_h(s) w_{k,h}}{\sum_{h=1}^{H} \varphi_h(s)}. \end{aligned} \tag{1}$$

Here, τ denotes the motion duration and α_x defines the decay rate of the phase variable s. $\boldsymbol{\xi}$, $\dot{\boldsymbol{\xi}}$ and $\ddot{\boldsymbol{\xi}}$ represent $\mathcal{O}$–dimensional position, velocity and acceleration, respectively. α_z and β_z regulate convergence dynamics, which are typically set to ensure critical damping (e.g., $\beta_z = \alpha_z/4$). $\mathbf{g}$ stands for the goal position. $f^k_{\mathbf{w}}(s)$ denotes the k–th element of $\mathbf{f}_{\mathbf{w}}(s)$, expressed as a linear combination of basis functions $\varphi_h(s)$. $\mathbf{w}$ represents the collection of all parameters $\{\{w_{k,h}\}_{h=1}^{H}\}_{k=1}^{K}$ and it captures the motion pattern of the demonstration. H is the number of basis functions.

B. Diffusion Policies

Diffusion policies learn action distributions via a forward diffusion process that perturbs actions with noise and a reverse denoising process that reconstructs noise-free actions conditioned on sensory inputs. These models can be parameterized in different but equivalent ways, for example by predicting the injected noise or directly predicting the clean-action estimate [18]–[20]. In this work, we adopt the clean-action parameterization. The main reason is that our action generation process is built on DMP, which is designed to represent smooth and structured motions rather than stochastic noise. In the forward process, a clean action sequence $\mathbf{A}^0$ is usually corrupted by Gaussian noise ϵ, i.e.,

$$\mathbf{A}^t = \mathbf{A}^0 + \epsilon, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \sigma_t^2 \mathbf{I}), \tag{2}$$

where σ_t is the noise scale at diffusion time t, and $\mathbf{I}$ denotes the identity matrix. The superscript t in the noisy action chunk $\mathbf{A}^t$ refers to continuous diffusion time.

When training the diffusion model, actions are perturbed via (2) with random-sampled noise scales. The model D_θ is then trained to minimize the reconstruction loss, i.e.,

$$D_\theta(\mathbf{A}^t, \mathbf{S}, \sigma_t) = \hat{\mathbf{A}}^0, \tag{3}$$

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{\sigma_t, \mathbf{A}^0, \epsilon}\big[\alpha(\sigma_t) \left\| D_\theta(\mathbf{A}^t, \mathbf{S}, \sigma_t) - \mathbf{A}^0 \right\|_2^2\big]. \tag{4}$$

Here, D_θ is used to predict a clean-action estimate $\hat{\mathbf{A}}^0$. $\alpha(\sigma_t)$ is a weighting function that adjusts the loss at different noise scales. $\mathbf{S}$ represents sensor observations, e.g., images and the robot's proprioceptive states.

The trained model generates new action sequences by solving a reverse-time probability flow ordinary differential equation (ODE) from an initial noisy sample $\mathbf{A}^T \sim \mathcal{N}(\mathbf{0}, \sigma_T^2 \mathbf{I})$ to $t = 0$, thereby deterministically transforming noisy actions back to clean ones [2], [3]:

$$\mathrm{d}\mathbf{A}^t = -\dot{\sigma}_t \sigma_t \nabla_{\mathbf{A}^t} \log p_t\left(\mathbf{A}^t \mid \mathbf{S}, \sigma_t\right) \mathrm{d}t, \tag{5}$$

$$\nabla_{\mathbf{A}^t} \log p_t(\mathbf{A}^t \mid \mathbf{S}, \sigma_t) \approx \frac{D_\theta(\mathbf{A}^t, \mathbf{S}, \sigma_t) - \mathbf{A}^t}{\sigma_t^2}. \tag{6}$$

Eq. (5) is numerically integrated backward in time using a finite-step solver (e.g., DDIM sampler [18] or EDM sampler [19]), yielding the final denoised action sequence $\mathbf{A}^0$. $\nabla_{\mathbf{A}^t} \log p_t\left(\mathbf{A}^t \mid \mathbf{S}, \sigma_t\right)$ denotes the score function, i.e., the gradient of the log-density of the noisy action distribution at diffusion time t. Since the score function is intractable, it is approximated by the learned model $D_\theta(\mathbf{A}^t, \mathbf{S}, \sigma_t)$ through (6). At each denoising step, D_θ predicts a clean-action estimate $\hat{\mathbf{A}}^0$ from the current noisy action chunk $\mathbf{A}^t$, and this estimate is used to update $\mathbf{A}^t$ towards a lower-noise sample $\mathbf{A}^{t-\Delta t}$ until $t \approx 0$.

IV. Robust Hybrid Diffusion

In this section, we introduce RhD, an imitation learning framework that unifies end-effector actions and joint actions (i.e., hybrid-space actions) for robotic manipulation, where goal-directed and perturbation-resilient actions in both Cartesian space and joint space are first generated through dynamical systems and diffusion models, and subsequently integrated using null-space control. An overview of the RhD framework is illustrated in Fig. 1.

We assume access to a multimodal dataset $\mathcal{D} = \{\mathbf{d}_n\}_{n=1}^N$ comprising N demonstrations. The n–th demonstration $\mathbf{d}_n$ consists of sensor observations $\mathbf{S}_t^n = \{\mathbf{O}_t^n, \mathbf{Q}_t^n, \mathbf{P}_t^n\}_{t=1}^{T(n)}$ and actions $\{\mathbf{A}_t^n\}_{t=1}^{T(n)}$. Here, the subscript t denotes the temporal index within a demonstration and $T(n)$ denotes the demonstration length. $\mathbf{O}_t^n$, $\mathbf{Q}_t^n$, and $\mathbf{P}_t^n$ represent image observations, joint angles, and end-effector poses, respectively. To capture temporal dependencies, observations and actions are organized in a chunked format: each observation $\mathbf{S}$ chunk contains the previous L observations, while each action chunk corresponds to the next M actions.

A. Diffusion-Enabled DMP Trajectory Generation

We handle both end-effector actions and joint actions separately and partition $\mathcal{D}$ into two subsets according to action type. The task space dataset is defined as $\mathcal{D}_{\text{eef}} = \{\{\mathbf{O}_t^n, \mathbf{P}_t^n, \mathbf{A}_{t,\text{eef}}^n\}_{t=1}^{T(n)}\}_{n=1}^N$, where $\mathbf{A}_{t,\text{eef}}^n$ denotes end-effector actions. Similarly, the joint space dataset is defined as $\mathcal{D}_{\text{jnt}} = \{\{\mathbf{O}_t^n, \mathbf{Q}_t^n, \mathbf{A}_{t,\text{jnt}}^n\}_{t=1}^{T(n)}\}_{n=1}^N$, where $\mathbf{A}_{t,\text{jnt}}^n$ represent joint actions. We then train two diffusion models, i.e., $D_{\theta_{\text{eef}}}$ and $D_{\theta_{\text{jnt}}}$, on $\mathcal{D}_{\text{eef}}$ and $\mathcal{D}_{\text{jnt}}$, respectively. When no ambiguity arises, we use D_θ as a generic notation for either model, since their main difference lies in whether the training actions are represented in task space or joint space.

To enable robust action generation, we embed DMP into the diffusion model in Eq. (3). In particular, we first use a deep neural network F_θ to predict DMP task parameters, i.e.,

$$[\hat{\mathbf{w}}, \hat{\mathbf{g}}] = F_\theta(\mathbf{A}^t, \mathbf{S}, \sigma_t). \tag{7}$$

Here, $\mathbf{A}^t \in \mathbb{R}^{M\times\mathcal{O}}$ denotes the noisy action chunk, where M is the action horizon and $\mathcal{O}$ denotes the dimension of the continuous action (e.g., end-effector pose or joint angle). $\mathbf{S}$ comprises image observation $\mathbf{O}$, end-effector pose $\mathbf{P}$, and joint angles $\mathbf{Q}$. σ_t represents the noise scale at diffusion time t. $\hat{\mathbf{w}} \in \mathbb{R}^{\mathcal{O}\times B}$ and $\hat{\mathbf{g}} \in \mathbb{R}^{\mathcal{O}}$ are DMP task parameters, representing the motion pattern and goal state within a local action chunk, respectively. B is the number of DMP basis functions. These predicted task parameters are then passed to DMP to generate the clean action estimate $\hat{\mathbf{A}}^0 \in \mathbb{R}^{M\times\mathcal{O}}$, i.e.,

$$\hat{\mathbf{A}}^0 = \rho(\mathbf{s}_c, \hat{\mathbf{w}}, \hat{\mathbf{g}}). \tag{8}$$

$\mathbf{s}_c$ denotes the current proprioceptive state (e.g., end-effector pose or joint angle) from sensor observations $\mathbf{S}$. In addition, we use ρ as a generic notation for either the Cartesian-space DMP ρ_{eef} or joint-space DMP ρ_{jnt} in (1). For ρ_{eef}, the three position coordinates and four quaternion coefficients are treated as independent scalar DMP channels, and each generated quaternion is normalized to unit length.

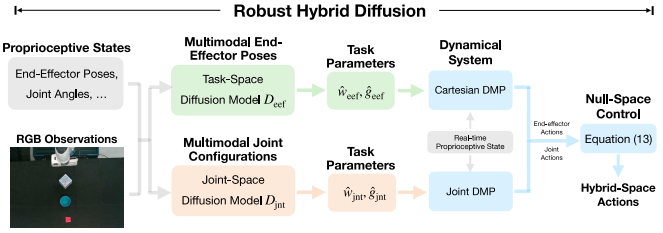


Fig. 1. An overview of the RhD framework. Task parameters in both Cartesian space and joint space are first predicted via two independent diffusion policies. Subsequently, the corresponding action sequences are generated using DMPs with these task parameters. Finally, the actions from both spaces are unified through the Jacobian-based null-space control.

The process of predicting a clean-action estimate can be expressed as

$$D_\theta(\mathbf{A}^t, \mathbf{S}, \sigma_t) = \rho(\mathbf{s}_c, F_\theta(\mathbf{A}^t, \mathbf{S}, \sigma_t)) = \rho(\mathbf{s}_c, \hat{\mathbf{w}}, \hat{\mathbf{g}}) = \hat{\mathbf{A}}^0. \tag{9}$$

Therefore, the diffusion training objective in Eq. (4) becomes

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{\sigma_t, \mathbf{A}^0, \epsilon}\Big[\alpha(\sigma_t)\big\|\rho\big(\mathbf{s}_c, F_\theta(\mathbf{A}^t, \mathbf{S}, \sigma_t)\big) - \mathbf{A}^0\big\|_2^2\Big]. \tag{10}$$

To enhance the model to generate accurate action chunks, we introduce two auxiliary losses: a goal loss $\mathcal{L}_{\text{goal}}$ to ensure precise convergence to the final-step action of each action chunk and a velocity loss $\mathcal{L}_{\text{vel}}$ to enforce a consistent motion profile, i.e.,

$$\begin{aligned} \mathcal{L}_{\text{goal}} &= \|\hat{\mathbf{g}} - \mathbf{g}\|_2^2, \\ \mathcal{L}_{\text{vel}} &= \left\|\dot{\hat{\mathbf{A}}}^0 - \dot{\mathbf{A}}^0\right\|_2^2, \end{aligned} \tag{11}$$

where $\mathbf{g}$ represents the local target (i.e., the final-step action) of the current action chunk. $\dot{\mathbf{A}}^0$ and $\dot{\hat{\mathbf{A}}}^0$ denote the temporal derivatives (via finite differences) of the ground-truth and predicted action chunks, respectively. The overall training objective combines the standard diffusion loss $\mathcal{L}_{\text{diff}}$ (see Eq. (4)) with the proposed auxiliary losses, i.e.,

$$\mathcal{L} = \lambda_1\mathcal{L}_{\text{diff}} + \lambda_2\mathcal{L}_{\text{goal}} + \lambda_3\mathcal{L}_{\text{vel}}, \tag{12}$$

where λ_1, λ_2, and λ_3 are weighting coefficients that balance the contributions of $\mathcal{L}_{\text{diff}}$, $\mathcal{L}_{\text{goal}}$, and $\mathcal{L}_{\text{vel}}$, respectively.

DMP allows for producing robust, goal-directed behavior that remains resilient under external disturbances. During inference, the DMP task parameters are first predicted by F_θ. These parameters are then held fixed until the robot reaches the

Algorithm 1 *Robust Hybrid Diffusion*

Require: Diffusion models $D_{\theta_{\text{eef}}}$ and $D_{\theta_{\text{jnt}}}$
Require: Noise scheduler $\sigma(t)$, distance metric $d(\cdot,\cdot)$
Inputs: $\mathbf{O}_t = \{\mathbf{o}_{t-L+l}\}_{l=1}^{L}$, $\mathbf{P}_t = \{\mathbf{p}_{t-L+l}\}_{l=1}^{L}$, $\mathbf{Q}_t = \{\mathbf{q}_{t-L+l}\}_{l=1}^{L}$
// Observations
Sample noisy end-effector actions $\mathbf{A}_{\text{eef}}^{T} \sim \mathcal{N}(\mathbf{0}, \sigma_T^2 \mathbf{I}_{\text{eef}})$
Sample noisy joint actions $\mathbf{A}_{\text{jnt}}^{T} \sim \mathcal{N}(\mathbf{0}, \sigma_T^2 \mathbf{I}_{\text{jnt}})$
Compute $\{\hat{\mathbf{w}}_{\text{eef}}, \hat{\mathbf{g}}_{\text{eef}}\}$ for Cartesian DMP ρ_{eef} via $D_{\theta_{\text{eef}}}(\mathbf{A}_{\text{eef}}^{T}, \mathbf{O}_t, \mathbf{P}_t, \sigma_T)$
// e.g., using DDIM sampler
Compute $\{\hat{\mathbf{w}}_{\text{jnt}}, \hat{\mathbf{g}}_{\text{jnt}}\}$ for joint DMP ρ_{jnt} via $D_{\theta_{\text{jnt}}}(\mathbf{A}_{\text{jnt}}^{T}, \mathbf{O}_t, \mathbf{Q}_t, \sigma_T)$
// e.g., using DDIM sampler
repeat
  $\mathbf{p}_d \leftarrow \rho_{\text{eef}}(\mathbf{p}_c, \hat{\mathbf{w}}_{\text{eef}}, \hat{\mathbf{g}}_{\text{eef}})$, where $\mathbf{p}_{\text{c}} = [\boldsymbol{\xi}_c \ \boldsymbol{\eta}_c]^{\top}$
  // Desired end-effector pose via DMP
  $\mathbf{q}_d \leftarrow \rho_{\text{jnt}}(\mathbf{q}_c, \hat{\mathbf{w}}_{\text{jnt}}, \hat{\mathbf{g}}_{\text{jnt}})$
  // Desired joint angles via DMP
  $\mathbf{J}_c \leftarrow \text{Jacobian}(\mathbf{q}_c)$
  // Compute Jacobian matrix
  Compute $\mathbf{q}^*$ via (13)
  // Compute hybrid-space action
  Move robot joints to $\mathbf{q}^*$
  Update $\mathbf{p}_c, \mathbf{q}_c$
until $d(\mathbf{p}_{\text{c}}, \hat{\mathbf{g}}_{\text{eef}}) \leq r$
// $r > 0$ is threshold
// Terminate when the task-space goal is reached

target $\hat{\mathbf{g}}$. The corresponding robot action chunk is generated by using ρ with $\mathbf{s}_c$.

B. Hybrid-Space Fusion via Null-Space Control

Although the predicted end-effector actions and joint actions offer robustness, they are defined in different action spaces (i.e., task space and joint space). Task-space actions directly specify the primary task-relevant end-effector motion, while joint-space actions represent demonstration-consistent whole-arm configurations. RhD combines these complementary advantages through null-space control [21], using the Cartesian prediction as the primary motion and the joint-space prediction as the secondary motion. Let's denote $\boldsymbol{\xi}_{\text{c}}$, $\boldsymbol{\xi}_d$, $\boldsymbol{\eta}_{\text{c}}$, and $\boldsymbol{\eta}_d$ as the current end-effector position, desired position, current quaternion, and desired quaternion, respectively. For a redundant robotic system, the hybrid space action $\mathbf{q}^*$ is given by

$$\mathbf{q}^* = \mathbf{q}_{\text{c}} + \mathbf{J}_c^{\dagger}\begin{pmatrix} \boldsymbol{\xi}_d - \boldsymbol{\xi}_{\text{c}} \\ 2\log(\boldsymbol{\eta}_d * \overline{\boldsymbol{\eta}}_{\text{c}}) \end{pmatrix} + (\mathbf{I} - \mathbf{J}_c^{\dagger}\mathbf{J}_c)(\mathbf{q}_d - \mathbf{q}_{\text{c}})\delta, \tag{13}$$

where $\mathbf{J}_c$ and $\mathbf{J}_c^{\dagger}$ denote the Jacobian matrix and its Moore-Penrose pseudoinverse at the current joint angle $\mathbf{q}_{\text{c}}$. $\mathbf{q}_d$ represents the desired joint angle. $\overline{\boldsymbol{\eta}}_c$ is the conjugation of $\boldsymbol{\eta}_c$. The function $\log(\cdot) : \mathbb{S}^3 \mapsto \mathbb{R}^3$ is used to determine the difference vector between two quaternions. $\delta > 0$ is a scalar that controls the influence of joint actions in null-space.

Equation (13) provides an analytical solution for exploiting kinematic redundancy. It ensures accurate tracking of the desired end-effector pose via Jacobian projection, while simultaneously enabling the expression of multimodal joint behaviors without interfering with task execution.

Diffusion-driven DMP trajectory generation and null-space control together constitute the proposed RhD framework. RhD first generates goal-directed actions in both task space and joint space using diffusion models coupled with DMPs, thereby improving robustness to external perturbations. The resulting actions are then integrated through (13), which enables effective exploitation of kinematic redundancy. Through this combination, RhD produces robust and multimodal end-effector motions and joint configurations while preserving task-level accuracy. The complete action generation pipeline is summarized in Algorithm 1. In practice, goal arrival is determined using the distance metric $d(\mathbf{s}_c, \hat{\mathbf{g}})$ (e.g., Euclidean metric), which measures the distance between the current robot state $\mathbf{s}_c$ and the target $\hat{\mathbf{g}}$. Once the arrival criterion is met, $D_{\theta_{\text{eef}}}$ and $D_{\theta_{\text{jnt}}}$ produce the next prediction.

V. EVALUATIONS

We now evaluate the effectiveness of the proposed RhD framework through experiments conducted in both simulated and real-world robotic settings. Our evaluations aim to answer the following key questions:

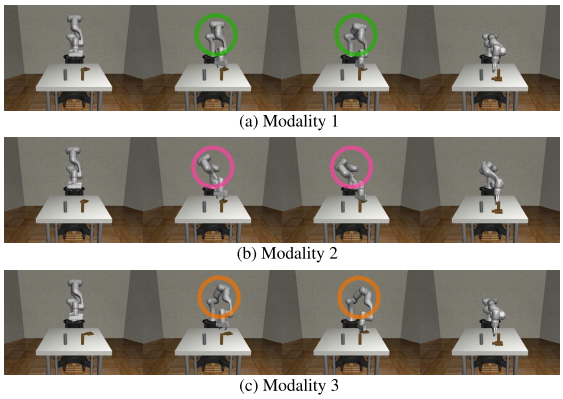
- Q1: What trade-offs arise between task-space diffusion policies and joint-space diffusion policies?
- Q2: To what extent does RhD mitigate these trade-offs?
- Q3: How does RhD compare with RK-Diffuser [6], a recent baseline for unified action generation?

In our experiments, we adopt the diffusion models in [1] as a baseline (denoted as DP). Specifically, the suffixes '-C' and '-T' denote DP variants implemented with U-Net and transformer architectures, respectively. We additionally use the tags '-EEF' and '-JNT' to indicate models trained exclusively in task space and joint space. These tags apply only to single-space policies. Since RK-Diffuser and RhD are hybrid-space methods that jointly leverage both end-effector actions and joint actions to generate hybrid-space actions, they are not labeled with '-EEF' or '-JNT'. To ensure fair comparisons, DP hyperparameters are kept consistent with the original settings in [1]. For RhD, we tune the DMP-specific hyperparameters and employ the DDIM sampler [18] for action generation. All models are trained until no further improvement in performance is observed.

A. Simulation Setup and Evaluation Metrics

The evaluations are conducted on the *Robomimic* benchmark [22], using three representative tasks: *Can*, *Square*, and *Lift*. The *Can* task requires picking up a can and placing it at a target location. The *Square* task involves grasping a square ring and inserting it onto a cylindrical peg. The *Lift* task consists of picking up an object from the table surface and lifting it to a specified height.

For each task, we additionally construct a multimodal dataset containing demonstrations with three distinct modalities in joint space. Across these demonstrations, the end-effector trajectories are similar, whereas the joint trajectories,



(a) Modality 1

(b) Modality 2

(c) Modality 3

Fig. 2. Three distinct patterns in joint space (highlighted with circles) for completing the *Square* task.

TABLE I
SUMMARY OF THE SIMULATED TASKS: #DOF: NUMBER OF DEGREES OF FREEDOM; #DEMO: NUMBER OF DEMONSTRATIONS FOR THE THREE MOTION MODALITIES, REPORTED AS M1/M2/M3, WHERE EACH ENTRY INDICATES THE NUMBER OF DEMONSTRATIONS BELONGING TO THE CORRESPONDING MODALITY; STEPS: MAXIMUM NUMBER OF ENVIRONMENT STEPS ALLOWED PER EVALUATION EPISODE; IMG: WHETHER USING IMAGE AS OBSERVATIONS; HIPREC: WHETHER THE TASK HAS A HIGH PRECISION REQUIREMENT

Task	#DoF	#Demo	Steps	Img	HiPrec
		M1/M2/M3			
Lift	7	60/70/70	400	Yes	No
Can	7	60/70/70	400	Yes	No
Square	7	185/93/134	400	Yes	Yes

particularly those of the joints near the base, exhibit clearly different motion patterns. An illustrative example of these patterns for the *Square* task is shown in Fig. 2, and a summary of the simulation setup is provided in Table I.

For each approach, we report both the best and average performance across the top five performing checkpoints, with each checkpoint evaluated over 100 randomized environment initializations. We use the following three metrics for evaluation: (1) **reward**, which measures the best and the average task success rate (with a maximum score 1.0); (2) **diversity**, which quantifies the distribution of distinct joint behavior patterns. The result a/b/c indicates that three unique patterns occurred a, b, and c times, respectively. Each generated trajectory is assigned automatically to one of these patterns using a fixed classification rule calibrated on manually labeled reference trajectories; and (3) **robust reward**, which assesses the best and the average task success rate when external disturbances are exerted on the robot arm. To impose disturbances, we apply a constant Cartesian force of 173N to the robot's end-effector for 2s, beginning 1 s after the start of each evaluation rollout, when the robot is approaching the manipulated object. The force is directed along the vector $[-1, -1, 1]^\top$, corresponding to a backward, leftward, and upward perturbation.

B. Evaluations on Simulation Tasks

1) Comparison between the task-space DP and joint-space DP: We begin by comparing the success rates of policies trained in task space and joint space. For the *Can* and the *Square* tasks, as reported in the reward columns of Table II, task-space DP (rows 1–2) consistently outperforms its joint-space counterpart (rows 3–4) by a large margin. For the *Lift* task, DP-C achieves perfect performance (100% success) in both action spaces. Overall, these results suggest that **task-space DP attains higher task success rates than joint-space DP**.

However, this improved task performance comes at the expense of joint behavior diversity. As shown in the diversity columns of Table II, the task-space DP exhibits highly monomodal joint behavior, with a single joint motion pattern dominating all 500 evaluations across all three tasks. For example, in the *Square* task, the joint behavior illustrated in Fig. 2(a) dominates all evaluations produced by task-space DP. In contrast, the **joint-space DP consistently reproduces all three distinct joint motion patterns**, as indicated by the diversity metric in rows 3–4. These findings highlight a fundamental trade-off between maximizing task success rate and preserving multimodal joint behaviors.

2) Comparison of RhD with the task-space and joint-space DPs: We first compare RhD (last row in Table II) with the task-space DP in terms of task success rate. Among all compared methods, task-space DP with the U-Net architecture (i.e., DP-C-EEF) achieves the highest success rates on all three tasks. RhD, however, remains highly competitive, trailing the best-performing model by only 2% on *Can* and 1% on *Square*, while achieving the same task performance on the *Lift* task. Note that RhD consistently outperforms DP-T-EEF across all tasks. These results suggest that **RhD attains task success rates comparable to those of task-space DP**.

We then compare RhD with the joint-space DP in terms of joint behavior diversity. The diversity metrics of RhD match those of the joint-space DP (rows 3–4), indicating that **RhD successfully reproduces multiple joint motion patterns**. Taken together, these results demonstrate that **RhD effectively balances strong task performance and multimodal joint behaviors**.

3) Comparison between RhD and RK-Diffuser: Finally, we compare RhD with RK-Diffuser. RK-Diffuser is a recent hybrid-space baseline that iteratively optimizes joint actions to match target end-effector poses. To evaluate its performance, we use a fixed update rate and vary the number of optimization iterations across three settings, i.e., 200, 600, and 1000. As shown in rows 5–7 and 8–10 of Table II, increasing the number of iterations generally leads to higher task success rates for RK-Diffuser.

On the *Square* and *Lift* tasks, RhD achieves the same best task success rates as the best-performing RK-Diffuser variant, i.e., RK-Diffuser-C ($N = 600$), and further attains a 2% higher average success rate on the *Square* task. On the *Can* task, RhD attains a 97% success rate, which is only 2% lower than that of the highest-performing RK-Diffuser variant, i.e., RK-Diffuser-C ($N = 200$). Although RhD and RK-Diffuser achieve similar task performance, their computation of the hybrid-space actions differs fundamentally. RK-Diffuser requires a post-hoc iterative optimization to compute hybrid-space actions after generating task-space and joint-space actions, whereas RhD

TABLE II

PERFORMANCE ON THE *Robomimic* BENCHMARK. WE REPORT REWARDS (I.E., SUCCESS RATES) IN THE FORMAT OF (MAXIMUM PERFORMANCE) / (AVERAGE PERFORMANCE) ACROSS THE BEST 5 CHECKPOINTS, WHERE EACH RESULT IS AVERAGED OVER 100 DIFFERENT INITIAL CONDITIONS FOR THE ENVIRONMENT. DIVERSITY REPRESENTS OCCURRENCES OF DIFFERENT JOINT MOTION PATTERNS ACROSS ALL EVALUATIONS. ROBUST REWARD ASSESSES (MAXIMUM PERFORMANCE) / (AVERAGE PERFORMANCE) ACROSS THE BEST 5 CHECKPOINTS IN THE PRESENCE OF DISTURBANCES

		Can			Square			Lift		
		Reward	Diversity	Robust Reward	Reward	Diversity	Robust Reward	Reward	Diversity	Robust Reward
Task Space	DP-T-EEF	0.92/0.91	500/0/0	0.45/0.41	0.90/0.89	500/0/0	0.14/0.11	**1.00/1.00**	500/0/0	0.19/0.05
	DP-C-EEF	**0.99/0.98**	500/0/0	0.51/0.47	**0.96/0.95**	500/0/0	0.24/0.17	**1.00/1.00**	500/0/0	0.82/0.69
Joint Space	DP-T-JNT	0.86/0.84	210/176/114	0.02/0.01	0.79/0.74	318/75/107	0.21/0.18	0.99/0.96	322/151/27	0.05/0.04
	DP-C-JNT	0.92/0.89	144/211/145	0.05/0.04	0.91/0.87	271/97/132	0.31/0.27	**1.00/1.00**	191/195/114	0.96/0.91
RK-Diffuser-T ($N = 200$)		0.92/0.90	235/181/84	0.34/0.30	0.90/0.87	320/69/111	0.22/0.18	**1.00**/0.99	283/170/47	0.35/0.16
RK-Diffuser-T ($N = 600$)		0.92/0.91	245/175/80	0.42/0.39	0.91/0.89	319/71/110	0.18/0.15	**1.00/1.00**	290/166/44	0.56/0.23
RK-Diffuser-T ($N = 1000$)		0.93/0.92	243/176/81	0.44/0.41	0.92/0.90	319/70/111	0.19/0.11	**1.00**/0.99	290/165/45	0.57/0.24
RK-Diffuser-C ($N = 200$)		**0.99**/0.97	169/210/121	0.10/0.09	0.92/0.89	261/130/109	0.19/0.16	**1.00/1.00**	204/205/91	0.90/0.84
RK-Diffuser-C ($N = 600$)		0.98/0.97	170/210/120	0.12/0.10	0.95/0.91	263/129/108	0.19/0.18	**1.00/1.00**	208/199/93	0.82/0.79
RK-Diffuser-C ($N = 1000$)		0.98/0.97	170/210/120	0.15/0.12	0.94/0.92	264/128/108	0.20/0.18	**1.00/1.00**	212/196/92	0.89/0.78
RhD		0.97/0.96	179/175/146	**0.95/0.93**	0.95/0.93	265/77/158	**0.94/0.89**	**1.00/1.00**	229/250/21	**0.99/0.97**

TABLE III

COMPONENT-ABLATION RESULTS ON THE *Can*, *Square*, AND *Lift* TASKS. THE BEST REWARD VALUES ARE SHOWN IN BOLD

	Can			Square			Lift		
	Reward	Diversity	Robust Reward	Reward	Diversity	Robust Reward	Reward	Diversity	Robust Reward
Det.-DMP-EEF	0.93/0.92	500/0/0	0.90/0.89	0.92/0.91	500/0/0	0.89/0.84	**1.00/1.00**	500/0/0	0.94/0.82
RhD w/o $D_{\theta_{\text{jnt}}}$	0.96/0.95	500/0/0	**0.96/0.95**	**1.00/0.97**	500/0/0	0.90/0.88	**1.00/1.00**	500/0/0	**1.00/0.98**
RhD w/o $D_{\theta_{\text{eef}}}$	0.88/0.83	202/184/114	0.81/0.80	0.80/0.77	257/85/158	0.66/0.60	0.97/0.96	190/287/23	0.89/0.86
DP-T-NS	0.93/0.92	242/177/81	0.40/0.37	0.93/0.91	323/68/109	0.18/0.15	**1.00**/0.99	293/163/44	0.47/0.21
RhD	**0.97/0.96**	179/175/146	0.95/0.93	0.95/0.93	265/77/158	**0.94/0.89**	**1.00/1.00**	229/250/21	0.99/0.97

avoids this time-consuming optimization and instead computes hybrid-space actions analytically via null-space control. A further comparison, including the computational time of hybrid-space actions and control latency, is provided in the real-world experiments.

We further observe that both DP and RK-Diffuser implemented with the U-Net architecture generally achieve higher task success rates than their transformer-based counterparts. This observation is consistent with the findings in [1], which showed that U-Net architectures usually yield more stable performance and are recommended as the default starting point for DP implementation. Notably, although RhD adopts a transformer architecture, it outperforms DP-T and RK-Diffuser-T while achieving performance comparable to DP-C and RK-Diffuser-C. These results suggest that the proposed integration of diffusion models with DMP-based action generation helps improve the effectiveness of transformer-based diffusion policies.

Another key advantage of RhD is its robustness to external disturbances. As shown in the robust reward columns of Table II, RhD maintains nearly identical task success rates relative to its nominal performance[1], with drops of only 2%, 1%, and 1% on the *Can*, *Square*, and *Lift* tasks, respectively. In contrast, RK-Diffuser experiences substantial performance degradation under the same conditions, with task success rates decreasing by nearly 90% on *Can*, 80% on *Square*, and 10% on *Lift*. One reason for this pronounced degradation is that RK-Diffuser does not explicitly incorporate real-time proprioceptive feedback during action-chunk execution. As a result, external perturbations can push the robot away from the training region, leading to off-distribution states from which subsequent action predictions become less reliable. This mismatch can accumulate over time and eventually result in task failure. **These findings highlight RhD's ability to generate robust actions that remain reliable under external disturbances**. Supplementary video further illustrates the performance of RhD in simulation, including its behaviour under external disturbances.

[1]We use "nominal" to represent evaluations without perturbations.

C. Ablation Study

We conducted component ablations on all three simulation tasks under the same evaluation protocol. In Table III, RhD w/o $D_{\theta_{\text{jnt}}}$ and RhD w/o $D_{\theta_{\text{eef}}}$ denote the task-space-only and joint-space-only variants, respectively; DP-T-NS denotes null-space fusion without DMPs; and Det.-DMP-EEF denotes the deterministic task-space DMP policy. RhD w/o $D_{\theta_{\text{jnt}}}$ achieves strong task performance but exhibits only one joint-motion modality, whereas RhD w/o $D_{\theta_{\text{eef}}}$ preserves all three modalities but generally achieves lower task performance. Full RhD combines the advantages of them. DP-T-NS retains nominal performance and joint-motion diversity but performs substantially worse than RhD under disturbances, justifying the application of DMPs. RhD w/o $D_{\theta_{\text{jnt}}}$ also outperforms Det.-DMP-EEF on all three tasks, showing the benefits of diffusion models compared to deterministic models under the same task-space DMP formulation.

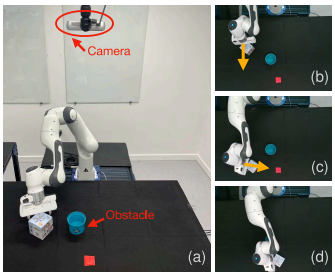


Fig. 3. The real-world box-pushing experiment. (*a*) Experimental setup. (*b*), (*c*), and (*d*) display the intermediate steps of accomplishing the task from the camera's view.



Fig. 4. Snapshots of the box-pushing task. Rows 1 to 4 show four distinct joint motion patterns generated by RhD, with the corresponding end-effector trajectories indicated by yellow arrows. The pink circles highlight key differences in joint configurations, demonstrating RhD's ability to produce diverse yet effective behaviors.

D. Evaluations on a Real-World Task

So far, we have demonstrated the effectiveness of RhD in simulation. We now extend our evaluations to a real-world robotic setting using a box-pushing task. In this task, the robot must push a box into a designated target zone (indicated by a red square marker) while avoiding collision with an obstacle. We collected 280 demonstrations using a 7-DoF Franka Emika Panda robotic arm and a top-mounted Intel RealSense D455 camera. These demonstrations comprise four distinct joint motion patterns used to accomplish the task (see Fig. 4), forming a multimodal dataset. Each demonstration consists of visual observations and the corresponding end-effector and joint trajectories recorded at 30 Hz. An overview of the experimental setup is shown in Fig. 3.

The objectives of this experiment are: (1) to compare the time required by RhD and RK-Diffuser to compute hybrid-space actions, and (2) to assess the robustness and safety of RhD when subjected to on-the-fly disturbances. We use RK-Diffuser-C with 1000 optimization iterations, as this configuration achieved the highest average success rate among all RK-Diffuser variants in simulation.

As shown in the third column of Table IV, because RK-Diffuser relies on post-hoc optimization of joint actions, it requires $5.743\,\mathrm{ms}$ to compute each hybrid-space action before sending it to the robot. This computational time restricts RK-Diffuser to controllers operating at relatively lower frequencies. In contrast, RhD computes hybrid-space actions analytically without iterative refinement and requires only $0.064\,\mathrm{ms}$ per control step, making it compatible with the high-frequency $1000\,\mathrm{Hz}$ controller used in our experiments.

TABLE IV
REAL-WORLD BOX-PUSHING EVALUATIONS. N/D DENOTES NOMINAL/DISTURBED SUCCESS RATES. DIVERSITY REPORTS THE FOUR MODALITY COUNTS OVER 75 NOMINAL TRIALS. ACTION-FUSION TIME IS REPORTED AS THE MEAN AND STANDARD DEVIATION PER HYBRID-SPACE ACTION. RECOVERY TIME IS REPORTED OVER SUCCESSFULLY RECOVERED TRIALS, AND PEAK ACCELERATION DENOTES THE MAXIMUM OBSERVED TRANSLATIONAL END-EFFECTOR ACCELERATION FOR THE PERTURBED EVALUATIONS

	N/D [%]	Diversity	Action Fusion Time [ms]	Recovery Time [s]	Peak Acceleration $[\mathrm{m/s^2}]$
RK-Diffuser	78.7/0.0	23/9/24/19	5.743 ± 0.149	–	5.93
RhD	82.7/76.7	25/20/9/21	0.064 ± 0.002	2.13 ± 0.36	1.73

Across three independent training seeds with 25 trials per seed, RhD achieves a pooled success rate of $62/75$ (82.7%), compared with $59/75$ (78.7%) for RK-Diffuser, as shown in the first column of Table IV, indicating similar nominal task performance within the evaluated sample size. Both methods preserve all four joint-motion modalities as shown in the second column of Table IV.

We further conducted 30 disturbed trials per method. The disturbance began when the robot first approached the box, at $12.67 \pm 1.82\,\mathrm{s}$ after execution started, and displaced the end-effector by $23.87 \pm 3.76\,\mathrm{cm}$ for $3.52 \pm 0.42\,\mathrm{s}$. The mean normalized displacement directions in the robot base frame were $[0.42, -0.90, 0.04]^\top$ and $[0.46, 0.87, 0.18]^\top$ for the two pushing directions. Recovery time was measured from disturbance release until the end-effector contacts the object. Representative disturbed executions of both methods are shown in Fig. 5.

As reported in the first column of Table IV, RhD recovered and completed $23/30$ disturbed trials (76.7%), whereas RK-Diffuser completed $0/30$ trials (0%). RhD achieved a mean recovery time of $2.13 \pm 0.36\,\mathrm{s}$. RK-Diffuser did not recover from the disturbances in any of the 30 trials; therefore, its recovery time is undefined, as denoted by "–" in Table IV. The maximum observed translational end-effector accelerations were $1.73\,\mathrm{m/s^2}$ for RhD and $5.93\,\mathrm{m/s^2}$ for RK-Diffuser, compared with $1.39\,\mathrm{m/s^2}$ under nominal execution. No collisions or joint-limit violations were observed for either method. These results demonstrate that RhD provides reliable disturbance recovery while maintaining its maximum acceleration substantially closer to the nominal level.

VI. CONCLUSIONS

In this work, we proposed RhD, a unified framework that integrates end-effector and joint actions for robust and versatile robotic manipulation. Through comprehensive evaluations in simulation and real-world tasks, we demonstrated that RhD achieves high task success rates while preserving multimodal joint motion patterns. Moreover, RhD efficiently computes hybrid-space actions without iterative refinement and exhibits reasonable robustness to external disturbances, highlighting its potential for broader real-world settings.

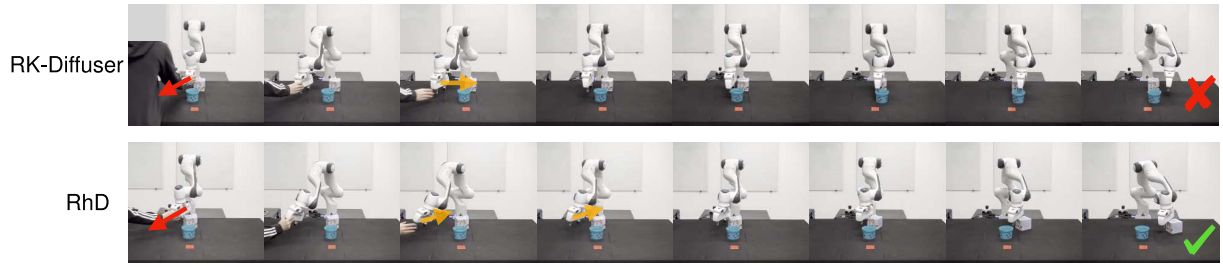


Fig. 5. Snapshots of the box-pushing task under external disturbances. *First and second rows* correspond to evaluations using RK-Diffuser and RhD, respectively. Red arrows indicate the direction of applied disturbances, while yellow arrows show the resulting action trajectories generated by each policy.

While RhD uses diffusion policies to predict DMP task parameters, the proposed framework can be integrated with alternative generative formulations, such as flow matching [23] or rectified flow models [24], to improve inference efficiency by reducing the number of sampling or integration steps required for action generation. Our current implementation adopts a standard DMP parameterization with a basis-function representation of the forcing term. Alternative parameterizations, such as directly predicting forcing-term values using Gaussian process regression or jointly learning the attractor-system gains under stability-preserving constraints, could also be incorporated, and their systematic evaluations are left for future work.

In addition, although DMP provides a compact and feedback-aware representation for goal-directed motion, their nonlinear forcing term vanishes as the phase variable decays. Consequently, when external disturbances occur near the end of execution, the vanishing forcing term has a trivial effect on trajectory-shape preservation, although the goal-attractor dynamics can still drive the motion toward the predicted goal within a local action chunk. A promising direction is to replace or augment DMPs with distance-field-based dynamical systems [25], which can encode the demonstrated motion geometry in a phase-independent manner and may better preserve the desired motion pattern under perturbations. The limitations of RhD also include the lack of reachability and collision checks, which is another important direction for future work.

REFERENCES

[1] C. Chi *et al.*, "Diffusion policy: Visuomotor policy learning via action diffusion," *Int. J. Robot. Res.*, vol. 44, no. 10–11, pp. 1684–1704, 2025.

[2] M. Reuss, M. Li, X. Jia, and R. Lioutikov, "Goal-conditioned imitation learning using score-based diffusion policies," in *Proc. Robot.: Sci. Syst.*, 2023.

[3] P. M. Scheikl *et al.*, "Movement primitive diffusion: Learning gentle robotic manipulation of deformable objects," *IEEE Robot. Autom. Lett.*, vol. 9, no. 6, pp. 5338–5345, Jun. 2024.

[4] T. Pearce *et al.*, "Imitating human behaviour with diffusion models," in *Proc. Int. Conf. Learn. Representations*, 2023.

[5] M. Aburub, C. C. Beltran-Hernandez, T. Kamijo, and M. Hamaya, "Learning diffusion policies from demonstrations for compliant contact-rich manipulation," in *Proc. IEEE/SICE Int. Symp. Syst. Integr.*, 2026, pp. 28–34.

[6] X. Ma, S. Patidar, I. Haughton, and S. James, "Hierarchical diffusion policy for kinematics-aware multi-task robotic manipulation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 18 081–18 090.

[7] Y. Liu, J. I. Hamid, A. Xie, Y. Lee, M. Du, and C. Finn, "Bidirectional decoding: Improving action chunking via guided test-time sampling," in *Proc. Int. Conf. Learn. Representations*, 2025.

[8] H. Kim, Y. Ohmura, and Y. Kuniyoshi, "Goal-conditioned dual-action imitation learning for dexterous dual-arm robot manipulation," *IEEE Trans. Robot.*, vol. 40, pp. 2287–2305, 2024.

[9] C. Wang *et al.*, "MimicPlay: Long-horizon imitation learning by watching human play," in *Proc. Conf. Robot. Learn.*, vol. 229. PMLR, 2023, pp. 201–221.

[10] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: Learning attractor models for motor behaviors," *Neural Comput.*, vol. 25, no. 2, pp. 328–373, 2013.

[11] D. Wang, C. Liu, F. Chang, and Y. Xu, "Hierarchical diffusion policy: Manipulation trajectory generation via contact guidance," *IEEE Trans. Robot.*, vol. 41, pp. 2086–2104, 2025.

[12] Z. Xian, N. Gkanatsios, T. Gervet, T.-W. Ke, and K. Fragkiadaki, "ChainedDiffuser: Unifying trajectory diffusion and keypose prediction for robotic manipulation," in *Proc. Conf. Robot. Learn.*, vol. 229. PMLR, 2023, pp. 2323–2339.

[13] X. Jia *et al.*, "MaIL: Improving imitation learning with selective state space models," in *Proc. Conf. Robot. Learn.*, vol. 270. PMLR, 2025, pp. 3888–3907.

[14] M. Reuss, J. Pari, P. Agrawal, and R. Lioutikov, "Efficient diffusion transformer policies with mixture of expert denoisers for multitask learning," in *Proc. Int. Conf. Learn. Representations*, 2025.

[15] H. Xue *et al.*, "Reactive diffusion policy: Slow-fast visual-tactile policy learning for contact-rich manipulation," in *Proc. Robot.: Sci. Syst.*, 2025.

[16] S. Wu, Y. Wang, and Y. Huang, "One-shot robust imitation learning for long-horizon visuomotor tasks from unsegmented demonstrations," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2025, pp. 8127–8133.

[17] G. Li, Z. Jin, M. Volpp, F. Otto, R. Lioutikov, and G. Neumann, "ProDMP: A unified perspective on dynamic and probabilistic movement primitives," *IEEE Robot. Autom. Lett.*, vol. 8, no. 4, pp. 2325–2332, Apr. 2023.

[18] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *Proc. Int. Conf. Learn. Representations*, 2021.

[19] T. Karras, M. Aittala, T. Aila, and S. Laine, "Elucidating the design space of diffusion-based generative models," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 26 565–26 577.

[20] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 6840–6851.

[21] Y. Huang, J. Silvério, L. Rozo, and D. G. Caldwell, "Hybrid probabilistic trajectory optimization using null-space exploration," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2018, pp. 7226–7232.

[22] A. Mandlekar *et al.*, "What matters in learning from offline human demonstrations for robot manipulation," in *Proc. Conf. Robot. Learn.*, vol. 164. PMLR, 2022, pp. 1678–1690.

[23] 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 *Proc. AAAI Conf. Artif. Intell.*, vol. 39, no. 14, 2025, pp. 14 754–14 762.

[24] R. Xue, J. Mao, M. Zhang, and Y. Wang, "RecFlow Policy: Fast and accurate visuomotor policy learning via rectified action flow," in *Proc. 7th Robot Learn. Workshop: Towards Robots with Human-Level Abilities at ICLR*, 2025.

[25] Y. Li and S. Calinon, "From movement primitives to distance fields to dynamical systems," *IEEE Robot. Autom. Lett.*, vol. 10, no. 9, pp. 9550–9556, Sep. 2025.