RoMAN-Flow: Taming Autoregressive Normalizing Flows for Offline Reinforcement Learning in Robotic Manipulation

Anonymous authors
Paper under double-blind review

Abstract

Offline reinforcement learning improves robotic policies using previously collected data without further environment interaction. Yet prevalent diffusion and flow-matching policies provide flexible continuous action distributions but do not offer tractable exact likelihoods, limiting their use in likelihood-based offline RL post-training. AR-NFs offer both expressive action modeling and exact likelihood evaluation, but their sequential sampling incurs substantial sampling overhead during policy optimization and deployment. We present RoMAN-Flow (Robotic Manipulation with Autoregressive Normalizing Flows), an offline reinforcement learning framework that makes AR-NF policies practical for robotic manipulation by addressing this sampling bottleneck in both stages. During policy optimization, RoMAN-Flow employs a sampling-free, advantage-weighted likelihood objective that assigns higher likelihood to high-advantage actions from the offline dataset without sampling from the autoregressive policy. For efficient deployment, it distills the optimized autoregressive policy into a one-step action generator, enabling low-latency action prediction. Experiments across simulated manipulation benchmarks and real-world robotic platforms demonstrate that RoMAN-Flow achieves competitive policy performance while substantially reducing inference latency.

1 Introduction

Vision-language-action (VLA) models have shown strong potential for robotic manipulation, yet pretrained policies often require further adaptation to downstream tasks. Offline reinforcement learning (RL) provides a practical approach to robotic policy post-training by improving policies from previously collected reward-labeled data without additional environment interaction.

Likelihood-based policy optimization has recently shown strong effectiveness in large language model (Achiam et al., 2023; Touvron et al., 2023) post-training, where autoregressive models provide tractable exact log-likelihoods for direct optimization on fixed datasets (Rafailov et al., 2023; Richemond et al., 2024). In robotic manipulation, however, this paradigm remains comparatively underexplored because the dominant diffusion (Chi et al., 2025) and flow-matching (Lipman et al., 2022) policies do not readily expose low-cost exact conditional action likelihoods. Autoregressive policies (Kim et al., 2024; Zitkovich et al., 2023) provide an alternative with tractable likelihoods, but they typically convert continuous robot actions into discrete token sequences. Such discretization introduces quantization error and may compromise fine-grained control by obscuring the continuous geometry and temporal coordination of robotic actions. Recent transformer-based autoregressive normalizing flows (AR-NFs) (Patacchiola et al., 2024; Kingma et al., 2016; Papamakarios et al., 2017) have demonstrated strong generative modeling capability in image generation (Zhai et al., 2025; Gu et al., 2026; Zheng et al., 2025; Zhao et al., 2025). By combining autoregressive dependency modeling with invertible transformations, AR-NFs substantially improve the expressiveness of conventional normalizing flows (Kobyzev et al., 2020; Dinh et al., 2017; 2014; Kolesnikov et al., 2024) while retaining tractable exact likelihood evaluation. These advances motivate us to investigate AR-NFs as a policy representation for likelihood-based offline robotic RL.

Despite these attractive properties, the autoregressive inverse of AR-NFs makes action generation sequential, which can be costly whenever policy sampling is required and directly limits deployment

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

efficiency. In this work, we present **RoMAN-Flow** (**Ro**botic **M**anipulation with **A**utoregressive **N**ormalizing Flows) an offline robotic reinforcement learning framework that exploits efficient likelihood evaluation during policy optimization and addresses sequential generation at deployment.

During offline RL post-training, RoMAN-Flow builds upon Implicit Q-Learning (IQL) (Kostrikov et al., 2021), whose actor update performs advantage-weighted maximum likelihood over actions contained in the offline dataset. Because this update does not require actions sampled from the current policy, the AR-NF only applies its parallel forward transformation to evaluate the exact likelihoods of dataset action chunks, avoiding costly autoregressive reverse sampling during reinforcement learning. RoMAN-Flow therefore increases the likelihood of high-advantage offline actions while retaining the expressive continuous action distribution modeled by the AR-NF policy.

To accelerate inference, we adapt one-step distillation (Zheng et al., 2025; Lu et al., 2025) to compress the trained RoMAN-Flow policy into a one-step actor. By matching the post-trained teacher's intermediate flow states and final action chunks, the one-step student learns to approximate its inverse transformation while generating all actions in parallel. The distilled policy substantially reduces inference latency by eliminating the sequential autoregressive reverse process, with only a modest performance drop across benchmarks.

We evaluate RoMAN-Flow on LIBERO (Liu et al., 2023), MetaWorld (Yu et al., 2020), RoboMimic (Mandlekar et al., 2021), and real-world robotic manipulation tasks. Across these settings, RoMAN-Flow achieves competitive policy performance while reducing policy inference latency by roughly $10\times$. These results demonstrate the potential of modern autoregressive normalizing flows as expressive, exact-likelihood policy models for offline robotic policy post-training.

Our main contributions are summarized as follows: (1) We introduce RoMAN-Flow, an offline robotic reinforcement learning framework based on autoregressive normalizing flows (AR-NFs), enabling expressive continuous-action modeling with tractable exact likelihood evaluation. (2) We exploit the forward–inverse asymmetry of AR-NFs for offline policy optimization: advantage-weighted likelihoods of dataset actions can be evaluated in parallel without invoking costly autoregressive sampling during actor updates. (3) We distill the optimized autoregressive policy into a one-step actor to reduce deployment latency, and validate the resulting optimization-to-deployment framework across multiple simulated benchmarks and real-world robotic tasks.

2 Related Work

2.1 Generative Policies for Robot Learning

Generative policies have become a dominant approach to robotic manipulation due to their expressive multimodal action distributions. Diffusion Policy introduced diffusion models for visuomotor control (Chi et al., 2025), followed by extensions to 3D observations and vision-language-action modeling (Ze et al., 2024; Wen et al., 2025). Flow matching provides an alternative generative formulation based on conditional vector fields (Lipman et al., 2022) and has been adopted by large-scale VLA models such as π_0 (Black et al., 2024) and $\pi_{0.5}$ (Physical Intelligence et al., 2025). However, diffusion and flow-matching policies do not expose low-cost exact action likelihoods: diffusion models rely on denoising objectives, while continuous flows require density tracking along the generation ODE (Ho et al., 2020; Song et al., 2020; Chen et al., 2018; Grathwohl et al., 2018). Their offline RL post-training therefore requires surrogate or specialized optimization procedures.

2.2 Offline RL and Normalizing-Flow Policies

Offline RL improves policies using fixed reward-labeled datasets while controlling distribution shift. Representative methods include behavior-regularized optimization (Fujimoto & Gu, 2021) and advantage-weighted policy extraction (Peng et al., 2019; Nair et al., 2020; Kostrikov et al., 2021). Likelihood-based offline policy optimization has also shown strong effectiveness in large language model post-training, where autoregressive models permit direct evaluation of policy likelihoods (Rafailov et al., 2023; Richemond et al., 2024). For diffusion policies, Diffusion-QL introduces value guidance during denoising (Wang et al., 2022), EDP adapts diffusion training to established offline RL objectives (Kang et al., 2023), and IDQL performs critic-guided extraction

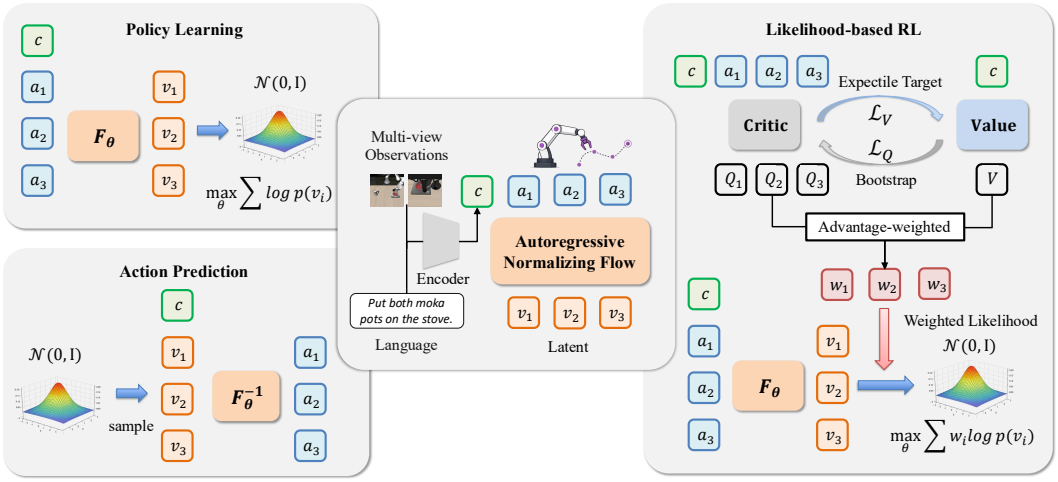


Figure 1: **Overview of RoMAN-Flow.** RoMAN-Flow models continuous action chunks with an invertible AR-NF conditioned on multimodal observations. Its forward transformation evaluates exact action likelihoods in parallel, enabling both likelihood-based imitation learning and advantage-weighted offline RL without autoregressive policy sampling. In contrast, action generation requires sequential autoregressive inversion from a Gaussian latent prior. RoMAN-Flow therefore distills the optimized AR-NF into a one-step policy for efficient deployment.

from a diffusion behavior model (Hansen-Estruch et al., 2023). These methods avoid direct exact-likelihood optimization through denoising surrogates, value guidance, or sampling-based extraction.

Normalizing flows (Kobyzev et al., 2020; Papamakarios et al., 2021; Mathieu & Nickel, 2020) instead provide exact density evaluation through invertible transformations (Rezende & Mohamed, 2015; Dinh et al., 2017), making them naturally compatible with likelihood-based policy optimization. Although conventional normalizing flows have been limited in expressiveness, recent autoregressive normalizing flows, including TarFlow (Zhai et al., 2025), STARFlow (Gu et al., 2026), and SimFlow (Zhao et al., 2025), demonstrate substantially improved generative modeling while retaining exact likelihoods. However, their autoregressive inversion leads to slow generation; FARMER (Zheng et al., 2025) and BiFlow (Lu et al., 2025) address a similar limitation through a one-step inverse. Building on these advances, we study AR-NFs as robotic policies, post-train them through likelihood-based offline RL, and distill the optimized policy for efficient deployment.

3 Method

We develop RoMAN-Flow, a likelihood-based offline robotic reinforcement learning framework based on autoregressive normalizing flow (AR-NF) policies. We first introduce the problem setup and conditional AR-NF policy, followed by likelihood-based offline RL post-training that increases the exact likelihood of high-advantage actions in the offline dataset without autoregressive policy sampling. We then detail one-step policy distillation for efficient deployment. An overview of our framework is shown in Figure 1.

3.1 Problem Setup

We consider imitation learning followed by offline RL post-training. Given a fixed dataset of reward-labeled trajectories, we organize each trajectory into overlapping length-H chunk transitions

$$x_{n,t} = (c_{n,t}, \mathbf{a}_{n,t}, \mathbf{r}_{n,t}, c_{n,t+H}), \; \mathcal{D}_H = \{x_{n,t}\}_{n,t}, \tag{1}$$

where $\mathbf{a}_{n,t} = (a_{n,t}, \ldots, a_{n,t+H-1})$ is a continuous action chunk and $\mathbf{r}_{n,t}$ contains the rewards. Imitation learning models the demonstrated action distribution, whereas offline RL uses the reward annotations to favor higher-value action chunks without additional environment interaction.

3.2 AUTOREGRESSIVE NORMALIZING FLOW POLICY

Given a chunk-level sample $(c_t, \mathbf{a}_t)$, RoMAN-Flow first encodes the visual observations, language instruction, and optional proprioceptive states into multimodal context tokens $\mathbf{C}_t$ using a pretrained encoder. Conditioned on context tokens $\mathbf{C}_t$, RoMAN-Flow then maps an observed action chunk $\mathbf{a}_t$ to a Gaussian latent $\mathbf{z}_t$ through an invertible AR-NF F_θ. The forward transformation provides exact conditional likelihoods for imitation learning and NF-IQL, whereas action generation applies the autoregressive inverse transformation F_θ^{-1} to a latent sampled from the Gaussian prior.

Multimodal context encoder. To fully exploit visual, linguistic, and proprioceptive conditions for action generation, we employ a pretrained vision-language model as the multimodal encoder. Given the policy condition c_t, the encoder produces context tokens $\mathbf{C}_t$. The context tokens are provided to every AR-NF flow block to condition action generation.

Autoregressive normalizing flow. We adopt SimFlow (Zhao et al., 2025) as the backbone and adapt it to model continuous robotic action chunks. The AR-NF transforms an action chunk $\mathbf{a}_t$ into a latent $\mathbf{z}_t$ through L conditional invertible flow blocks $F_\theta = f_L \circ f_{L-1} \circ \cdots \circ f_1$.:

$$\mathbf{h}_t^{(0)} = \mathbf{a}_t, \qquad \mathbf{h}_t^{(l)} = f_l\left(\mathbf{h}_t^{(l-1)}; \mathbf{C}_t\right), \qquad \mathbf{z}_t = \mathbf{h}_t^{(L)}, \tag{2}$$

where $\mathbf{h}_t^{(l)}$ denotes the output of the l-th AR-NF block.

Each block employs a prefix-causal Transformer. At action position j, the Transformer predicts an affine shift $\boldsymbol{\mu}_{t,j}^{(l)}$ and log-scale $\mathbf{s}_{t,j}^{(l)}$ using the multimodal context and preceding action positions, and the corresponding forward transformation is

$$\left(\boldsymbol{\mu}_{t,j}^{(l)}, \mathbf{s}_{t,j}^{(l)}\right) = T_l\left(\mathbf{C}_t, \mathbf{h}_{t,<j}^{(l-1)}\right), \qquad \mathbf{h}_{t,j}^{(l)} = \left(\mathbf{h}_{t,j}^{(l-1)} - \boldsymbol{\mu}_{t,j}^{(l)}\right) \odot \exp\left(-\mathbf{s}_{t,j}^{(l)}\right). \tag{3}$$

Because the affine parameters at position j depend only on $\mathbf{C}_t$ and preceding positions, each flow block has a triangular Jacobian. The exact conditional log-likelihood is therefore

$$\log \pi_\theta\left(\mathbf{a}_t \mid c_t\right) = \log p_0\left(\mathbf{z}_t\right) - \sum_{l=1}^{L}\sum_{j=1}^{H} \mathbf{s}_{t,j}^{(l)}, \tag{4}$$

where $p_0(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ is the standard Gaussian density.

Imitation learning and action generation. We initialize RoMAN-Flow by minimizing the negative log-likelihood of demonstrated action chunks:

$$\mathcal{L}_{\mathrm{IL}} = -\mathbb{E}_{(c_t, \mathbf{a}_t)\sim\mathcal{D}_H}\left[\log \pi_\theta\left(\mathbf{a}_t \mid c_t\right)\right]. \tag{5}$$

Because the complete action chunk is observed during training, the affine parameters for all positions can be evaluated in parallel through a single masked Transformer forward pass. This enables efficient exact-likelihood optimization during both imitation learning and NF-IQL post-training.

At inference time, the policy samples $\mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and generates an action chunk through

$$\mathbf{a}_t = F_\theta^{-1}\left(\mathbf{z}_t; \mathbf{C}_t\right), \qquad F_\theta^{-1} = f_1^{-1} \circ f_2^{-1} \circ \cdots \circ f_L^{-1}. \tag{6}$$

Within each inverse block, the affine parameters depend on previously recovered action positions, requiring the action chunk to be reconstructed sequentially. This forward–inverse asymmetry provides parallel exact-likelihood evaluation during training but introduces substantial latency at inference.

3.3 LIKELIHOOD-BASED OFFLINE POLICY OPTIMIZATION

To improve the behavior-cloned RoMAN-Flow policy beyond imitation learning, we build upon Implicit Q-Learning (IQL) (Kostrikov et al., 2021) and instantiate its advantage-weighted policy extraction with the exact AR-NF likelihood, which we refer to as NF-IQL. Conventional actor-critic methods such as TD3+BC (Fujimoto & Gu, 2021) require evaluating the critic on actions generated by the current policy. For an AR-NF actor, this invokes sequential inverse transformation and incurs

substantial training overhead, as shown in Table 13. In contrast, the IQL actor update uses only actions from the offline dataset. It therefore requires the parallel AR-NF forward transformation for exact likelihood evaluation, avoiding autoregressive policy sampling during actor optimization.

NF-IQL consists of two coupled components. First, it learns chunk-level action values and a state-value baseline from the offline data. Second, it converts their difference into an advantage weight and uses this weight to optimize the exact conditional likelihood of each dataset action chunk.

Chunk-level Q and state-value learning. We jointly learn an ensemble of M Chunk Transformer critics $\{Q_{\omega_m}\}_{m=1}^{M}$ and a state-value network V_ϕ, where ω_m and ϕ denote their trainable parameters. For stable value estimation, we maintain a target copy $\{Q_{\bar{\omega}_m}\}_{m=1}^{M}$ of the online critic ensemble, where $\bar{\omega}_m$ is updated from ω_m through Polyak averaging and receives no gradient updates. Given an action chunk, each critic Q_{ω_m} outputs Q-value estimates for all action prefixes, while V_ϕ maps the policy condition c_t to a scalar state value $V_\phi(c_t)$. The state value serves both as the bootstrap target for critic learning and as the state-dependent baseline for subsequent policy improvement.

Following CO-RFT (Huang et al., 2025), the m-th critic produces prefix-level Q-value estimates in a single causally masked forward pass: $\{Q_{\omega_m}^{(j)}(c_t, a_{t:t+j})\}_{j=0}^{H-1},\ m = 1, \ldots, M$, where j indexes the action prefix and $a_{t:t+j} = (a_t, \ldots, a_{t+j})$. Each action token attends to the policy condition and preceding actions while being prevented from accessing future actions.

For the j-th action prefix, we construct a temporal-difference target using the discounted rewards accumulated within the prefix and the bootstrapped value of the subsequent state:

$$y_{t,j} = \sum_{i=0}^{j} \gamma^i r_{t+i} + \gamma^{j+1}(1 - d_{t+j})\,\mathrm{sg}[V_\phi(c_{t+j+1})]\,, \tag{7}$$

where $\gamma \in (0, 1]$ is the discount factor, d_{t+j} indicates episode termination, and $\mathrm{sg}[\cdot]$ denotes stop-gradient. The online critics regress all prefix-level Q-values toward temporal-difference targets:

$$\mathcal{L}_Q = \mathbb{E}_{\mathcal{D}_H}\left[\frac{1}{MH}\sum_{m=1}^{M}\sum_{j=0}^{H-1}\left(Q_{\omega_m}^{(j)}\left(c_t, a_{t:t+j}\right) - y_{t,j}\right)^2\right]. \tag{8}$$

For policy weighting, we aggregate the target-critic estimates across action prefixes and ensemble members into a scalar critic score:

$$\bar{Q}_{\bar{\omega}}\left(c_t, \mathbf{a}_t\right) = \frac{1}{MH}\sum_{m=1}^{M}\sum_{j=0}^{H-1} Q_{\bar{\omega}_m}^{(j)}\left(c_t, a_{t:t+j}\right), \tag{9}$$

where $\bar{\omega}_m$ denotes the parameters of the m-th target critic. This aggregation is used as a multi-prefix critic score for policy weighting rather than as an unbiased estimate of a single fixed-horizon return; we ablate the aggregation choice against the final-prefix estimate in section C. The state-value network is optimized via expectile regression over the aggregated critic scores:

$$\mathcal{L}_V = \mathbb{E}_{(c_t, \mathbf{a}_t)\sim\mathcal{D}_H}\left[\rho_\tau\left(\mathrm{sg}\left[\bar{Q}_{\bar{\omega}}\left(c_t, \mathbf{a}_t\right)\right] - V_\phi(c_t)\right)\right], \tag{10}$$

where $\rho_\tau(u) = |\tau - \mathbb{I}[u < 0]|u^2$ and $\tau \in (0, 1)$ is the expectile parameter. For $\tau > 0.5$, this objective yields an upper-expectile state-value baseline.

Advantage-weighted likelihood optimization. We define the advantage of each offline action chunk and its corresponding likelihood weight as

$$A_t = \bar{Q}_{\bar{\omega}}\left(c_t, \mathbf{a}_t\right) - V_\phi(c_t), \qquad w_t = \exp\left(\beta A_t\right), \tag{11}$$

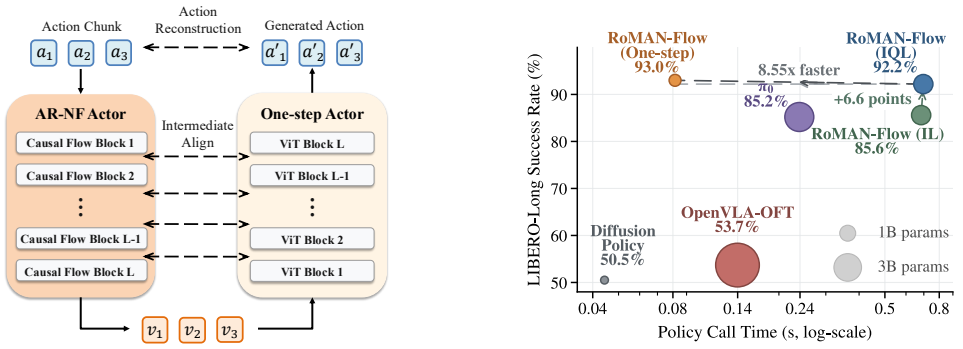
where $\beta > 0$ controls the strength of advantage weighting. Action chunks whose estimated values exceed the state-value baseline receive larger weights.

Following the IQL actor update, we post-train the AR-NF by minimizing the advantage-weighted negative log-likelihood:

$$\mathcal{L}_\pi = -\mathbb{E}_{(c_t, \mathbf{a}_t)\sim\mathcal{D}_H}\left[\mathrm{sg}\left[w_t\right]\log \pi_\theta\left(\mathbf{a}_t \mid c_t\right)\right]. \tag{12}$$

The target-critic estimate, state value, advantage, and likelihood weight are treated as fixed during the actor update. Because $\mathbf{a}_t$ comes directly from the offline dataset, this actor update requires only exact forward likelihood evaluation and never invokes autoregressive policy sampling.

Training pipelines, reward relabeling, and target-network updates are detailed in the Appendix A.



(a) One-step policy distillation.

(b) Speed–performance trade-off on LIBERO-Long.

Figure 2: One-step policy distillation and efficient inference. (a) The student aligns with the teacher's intermediate reverse states and reconstructs the final action chunk in a single forward pass. (b) Speed–performance trade-off on LIBERO-Long with action horizon $H = 16$. Each point reports the average success rate and action-chunk generation latency over the ten LIBERO-Long tasks, with marker area proportional to the number of inference-time policy parameters.

3.4 One-Step Policy Distillation

Although likelihood-based post-training avoids autoregressive policy sampling, deploying the optimized AR-NF still requires sequential inverse transformation. To accelerate policy execution, we freeze the post-trained AR-NF as a teacher and distill its autoregressive inverse mapping into a bidirectional student that generates an entire action chunk in one forward pass.

Data-induced trajectory distillation. As shown in Figure 2a, we instantiate the student g_ψ as a bidirectional Transformer that processes all action positions in parallel. Given an offline action chunk $\mathbf{a}_t$, we apply a small Gaussian perturbation, $\widetilde{\mathbf{a}}_t = \mathbf{a}_t + \boldsymbol{\epsilon}_t$ with $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$. The frozen teacher maps $\widetilde{\mathbf{a}}_t$ to a latent code $\mathbf{z}_t$ and records intermediate states $\{\mathbf{h}_t^{(l)}\}_{l=0}^{L-1}$. Conditioned on the same multimodal context $\mathbf{C}_t$, the student predicts complete action chunk and its intermediate states:

$$\left(\widehat{\mathbf{u}}_t^{(1)}, \ldots, \widehat{\mathbf{u}}_t^{(L)}, \widehat{\mathbf{a}}_t\right) = g_\psi\left(\mathbf{z}_t; \mathbf{C}_t\right). \tag{13}$$

We train the student through intermediate-state alignment and action-chunk reconstruction:

$$\mathcal{L}_{\text{data}} = \frac{\lambda_s}{L}\sum_{r=1}^{L}\left\|\widehat{\mathbf{u}}_t^{(r)} - \mathbf{h}_t^{(L-r)}\right\|_2^2 + \lambda_a\left\|\widehat{\mathbf{a}}_t - \mathbf{a}_t\right\|_2^2, \tag{14}$$

where λ_s and λ_a balance intermediate-state alignment and action reconstruction, respectively.

Prior-sampled trajectory distillation. The data-induced branch covers only latent codes obtained from offline action chunks. To expose the student to a broader portion of the post-trained teacher distribution, we additionally introduce prior-sampled distillation. For each policy condition, we sample latent $\mathbf{z}_t^p$ and decode it through the frozen NF-IQL teacher, parameterized by $\theta^\star$:

$$\mathbf{z}_t^p \sim \mathcal{N}(\mathbf{0}, \mathbf{I}),\ \left(\mathbf{a}_t^p, \{\mathbf{h}_t^{p,(l)}\}_{l=0}^{L-1}\right) = F_{\theta^\star}^{-1}\left(\mathbf{z}_t^p; \mathbf{C}_t\right). \tag{15}$$

Applying the same intermediate-state alignment and reconstruction objective to teacher-generated trajectories defines the prior-sampling loss $\mathcal{L}_{\text{prior}}$. The complete distillation objective is $\mathcal{L}_{\text{distill}} = \mathcal{L}_{\text{data}} + \lambda_p\mathcal{L}_{\text{prior}}$, where λ_p denotes the prior-sampling loss weight.

4 Experiments

4.1 Experimental Setup

Benchmarks. We evaluate RoMAN-Flow on MetaWorld-MT50 (Yu et al., 2020), LIBERO (Liu et al., 2023), RoboMimic MH (Mandlekar et al., 2021). MetaWorld-MT50 contains 50 manipulation

Table 1: **Controlled experiments on RoboMimic MH.** (a) Main results over four training seeds with 100 evaluation rollouts per seed. (b) Prefix-critic aggregation on Square-MH. (c) Distillation-source ablation on Square-MH. SR denotes success rate (%).

(a) RoboMimic MH

Method	Lift	Can	Square
SERNF (IL)	79 ± 0	96 ± 0	61 ± 4
SERNF (TD3+BC)	91 ± 4	96 ± 2	68 ± 5
Gaussian (IL)	$\mathbf{100 \pm 0}$	96 ± 1	74 ± 5
Gaussian (NF-IQL)	$\mathbf{100 \pm 0}$	95 ± 1	78 ± 6
RoMAN-Flow (IL)	$\mathbf{100 \pm 0}$	$\mathbf{97 \pm 2}$	80 ± 3
RoMAN-Flow (NF-IQL)	$\mathbf{100 \pm 0}$	96 ± 3	$\mathbf{85 \pm 3}$
RoMAN-Flow (One-Step)	$\mathbf{100 \pm 0}$	95 ± 2	80 ± 3

(b) Prefix aggregation

Aggregation	Episodes	SR
Last	100	87.0
Mean	100	88.0

(c) Distillation source

Source	Episodes	SR
Data-only	100	84.0
Full	100	**88.0**

Table 2: Success rates (%) on MetaWorld-MT50. Avg. is the unweighted mean over the four difficulty groups. Bold and underlined numerical entries denote the best and second-best results.

Method	#Params	Easy	Medium	Hard	Very Hard	Average
Diffusion Policy (Chi et al., 2025)	157M	23.1	10.7	1.9	6.1	10.5
TinyVLA (Wen et al., 2025)	1.3B	77.6	21.5	11.4	15.8	31.6
SmolVLA (Shukor et al., 2025)	2.25B	<u>87.1</u>	51.8	70.0	64.0	68.2
π_0 (Black et al., 2024)	3.3B	77.9	51.8	53.3	20.0	50.8
π_0 + Flow-SDE (Chen et al., 2025)	3.3B	**92.1**	**74.6**	61.7	**84.0**	78.1
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	2.3B	68.2	37.3	41.7	28.0	43.8
$\pi_{0.5}$ + Flow-SDE (Chen et al., 2025)	2.3B	86.4	55.5	75.0	66.0	70.7
RoMAN-Flow (IL)	1.45B	80.4	63.6	75.0	72.0	72.8
RoMAN-Flow (NF-IQL)	1.45B	80.0	<u>74.5</u>	**90.0**	<u>80.0</u>	**81.1**
RoMAN-Flow (One-Step)	0.56B	81.4	72.7	<u>80.0</u>	<u>80.0</u>	<u>78.5</u>

tasks spanning four difficulty groups and evaluates large-scale multitask learning. LIBERO consists of four ten-task suites–Spatial, Object, Goal, and Long–covering spatial, object, goal-conditioned, and long-horizon generalization, respectively. On RoboMimic MH, we follow the training and evaluation protocol of SERNF (Yang et al., 2026) on Lift, Can, and Square.

Real-world evaluation. We evaluate RoMAN-Flow on the Franka–XHand platform shown in Figure 3, consisting of a seven-DoF Franka arm and a twelve-DoF XHand dexterous hand. We consider four tasks: Pick Beaker, Pick Cylinder, Place Beaker, and Put Beaker on Balance.

We compare three stages of RoMAN-Flow: imitation learning, denoted as RoMAN-Flow (IL); NF-IQL post-training, denoted as RoMAN-Flow (NF-IQL); and the distilled policy, denoted as RoMAN-Flow (One-Step). Unless otherwise stated, comparisons among RoMAN-Flow variants use the same offline data, observation encoder, action horizon, and evaluation protocol. Results on MetaWorld and LIBERO are included to provide benchmark context; because these methods differ in model backbone, pretraining, and optimization protocol, we rely on the matched RoboMimic comparisons below for component-level attribution. Implementation are detailed in the Appendix B.2.

4.2 Controlled Analysis of Policy Parameterization and Offline RL

We conduct controlled experiments on RoboMimic MH to isolate the effects of policy parameterization and offline RL optimization before considering comparisons across heterogeneous models.

Effect of policy parameterization. Table 1 provides matched comparisons between a diagonal Gaussian actor and the AR-NF actor used by RoMAN-Flow. We replace only the actor parameterization while keeping the encoder, offline dataset, critic, IQL objective, and training and evaluation protocol fixed. On the challenging Square task, IQL improves the Gaussian actor from 74 ± 5 to 78 ± 6, whereas the AR-NF improves from 80 ± 3 to 85 ± 3. Lift is already saturated and Can remains nearly unchanged after post-training. Under the same likelihood-based policy update, the AR-NF therefore achieves higher final performance than the Gaussian actor, most clearly on Square (85% vs. 78%). This comparison indicates that tractable likelihood alone is not sufficient: the more expres-

Table 3: Success rates (%) on the four LIBERO task suites. Average is the unweighted mean across suites. Bold and underlined numerical entries denote the best and second-best results, respectively.

Method	#Params	Spatial	Object	Goal	Long	Average
Diffusion Policy (Chi et al., 2025)	157M	78.3	92.5	68.3	50.5	72.4
OpenVLA (Kim et al., 2024)	7.5B	84.7	88.4	79.2	53.7	76.5
NORA-Long (Hung et al., 2025)	3B	92.2	95.4	89.4	74.6	87.9
SmolVLA (Shukor et al., 2025)	2.25B	93.0	94.0	91.0	77.0	88.8
GR00T-N1 (Bjorck et al., 2025)	2.2B	94.4	97.6	93.0	90.6	93.9
π_0 (Black et al., 2024)	3.3B	**96.8**	98.8	**95.8**	85.2	94.2
UniVLA (Full) (Bu et al., 2025)	$\sim$7.5B	<u>96.5</u>	96.8	<u>95.6</u>	92.0	<u>95.2</u>
RoMAN-Flow (IL)	1.45B	95.0	<u>99.2</u>	94.2	85.6	93.5
RoMAN-Flow (NF-IQL)	1.45B	95.0	**99.4**	94.6	<u>92.2</u>	**95.3**
RoMAN-Flow (One-Step)	0.56B	94.4	95.6	91.8	**93.0**	93.7

Table 4: Success rates (%) on four real-robot manipulation tasks. Average denotes the unweighted mean across the four tasks.

Method	Pick_beaker	Pick_cylinder	Place_beaker	Balance	Average
Diffusion Policy (Chi et al., 2025)	7	0	0	0	1.5
π_0 (Black et al., 2024)	80	3	93	73	62.3
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	93	10	97	87	71.8
RoMAN-Flow (IL)	37	20	**100**	73	57.3
RoMAN-Flow (NF-IQL)	**100**	**33**	**100**	93	**81.5**
RoMAN-Flow (One-Step)	80	20	80	**100**	69.5

sive AR-NF parameterization provides additional benefits under the same offline policy extraction procedure. We additionally report SERNF (Yang et al., 2026) as a prior NF offline RL baseline. Because SERNF differs from RoMAN-Flow in both flow architecture and offline RL algorithm, we treat it as a benchmark reference rather than using it for component-level attribution.

Efficiency of offline RL optimization. A central computational motivation for combining AR-NFs with IQL is that the IQL actor update evaluates only offline actions and therefore avoids the sequential AR-NF inverse during policy optimization. To directly quantify this effect, we compare TD3+BC and IQL on RoboMimic Square-MH using the same AR-NF initialization, dataset, critic, hardware, and batch size. As shown in Table 9, after warm-up, IQL reduces the update time from 7.27 s to 0.56 s, increases throughput from 2.20 to 28.6 samples/s, and reduces peak GPU memory from 56.3 to 17.8 GB. These results directly verify the computational advantage of dataset-action policy extraction for AR-NF post-training.

4.3 Policy Improvement Across Diverse Benchmarks

After isolating policy parameterization and optimization, we next evaluate whether offline post-training improves the same RoMAN-Flow policy across diverse simulation and real-world settings.

Large-scale multitask evaluation on MetaWorld. Table 2 reports success rates on MetaWorld-MT50. The primary comparison is between different stages of the same RoMAN-Flow policy. NF-IQL increases the average success rate from 72.8% to 81.1% (+8.3 points), with gains concentrated in the Medium, Hard, and Very Hard groups, while the already strong Easy performance remains essentially unchanged. The one-step student achieves 78.5%, corresponding to 96.8% of the NF-IQL teacher's average success rate. Results from existing methods are included to provide benchmark context rather than controlled component-level comparisons.

Multi-dimensional task generalization on LIBERO. Table 3 reports results on the four LIBERO suites. NF-IQL increases the mean success rate from 93.5% to 95.3%, with the gain concentrated on LIBERO-Long, where success increases from 85.6% to 92.2% (+6.6 points). Performance on Spatial, Object, and Goal remains close to the already strong imitation-learning policy, suggesting substantial saturation on these suites. The larger improvement on LIBERO-Long indicates that offline post-training is particularly beneficial in the less saturated, long-horizon setting. The one-step student achieves 93.0% on LIBERO-Long.

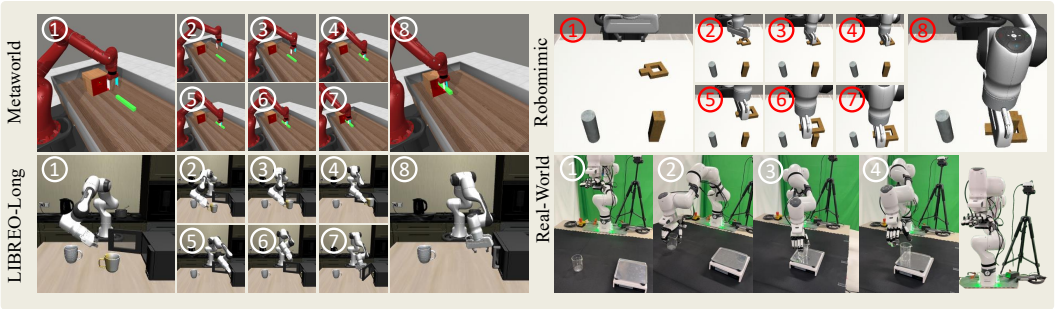


Figure 3: Qualitative results of RoMAN-Flow on MetaWorld, LIBERO, RoboMimic, and real-world tasks. Each row shows selected frames from a rollout, while the bottom-right inset presents our real-robot platform with a seven-DoF Franka arm and a twelve-DoF XHand dexterous hand.

Real-world manipulation. Table 4 reports success rates on the real-robot tasks. Starting from the same imitation-learned policy, NF-IQL increases the average success rate from 57.3% to 81.5% (+24.2%), with the largest improvements on Pick Beaker, Pick Cylinder, and Put Beaker on Balance. These results show that offline reward information can substantially refine grasping and precise placement behavior under visual variation and control noise. The one-step student achieves 69.5%, remaining above the IL policy but exhibiting a larger distillation gap than observed in simulation.

4.4 One-Step Distillation and Efficient Deployment

The optimized AR-NF still requires sequential inversion at deployment, so we distill it into a bidirectional one-step student for parallel action generation. The student retains 96.8% and 97.9% of teacher performance on MetaWorld and RoboMimic, slightly improves LIBERO-Long from 92.2% to 93.0%, and retains 85.3% on real-robot tasks. On LIBERO-Long, the student reduces action-generation latency from 697 ms to 81.5 ms, yielding an $8.55\times$ end-to-end speedup (Figure 2b). As the student is also smaller than the teacher (0.56B vs. 1.45B), the speedup reflects both one-step generation and model compression.

4.5 Component Analysis

Prefix aggregation. We compare the default mean aggregation of prefix-level critics with using only the final-prefix estimate on RoboMimic Square-MH. Table 7 shows that the two variants achieve similar success rates, indicating that performance is not sensitive to this aggregation choice.

Distillation source. Removing prior-sampled supervision from the full distillation objective reduces Square success from 88% to 84%, showing its complementary benefit beyond data-induced supervision and the importance of diverse teacher trajectories for effective distillation.

Qualitative results. Figure 3 shows representative rollouts on MetaWorld, LIBERO-Long, RoboMimic, and real-world tasks. RoMAN-Flow exhibits coherent multi-stage behaviors, including object interaction, transport, and precise target placement. More results are shown in Appendix D.

5 Conclusion

We introduced RoMAN-Flow, a likelihood-based offline reinforcement learning framework tailored for autoregressive normalizing flow policies in robot manipulation. RoMAN-Flow addresses the two main challenges of AR-NFs: NF-IQL avoids costly autoregressive sampling during post-training, while one-step distillation removes sequential inversion at deployment. Experiments on simulation and real-robot benchmarks demonstrate effective policy improvement and an $8.55\times$ inference speedup with limited performance loss. Our work establishes AR-NFs as a practical policy family for likelihood-based offline robotic reinforcement learning.

AI USE STATEMENT

We used OpenAI Codex, a generative AI tool, to assist with English-language editing, code debugging, and the generation of figures and tables. The datasets used in this work consist entirely of existing public benchmarks and datasets collected manually by the authors. No generative AI tools were used to create, fabricate, or alter the datasets or experimental results. All AI-assisted language and code were manually reviewed by the authors, and the generated figures and tables were checked for correctness and consistency. The authors take full responsibility for the final content of this paper, including all text, claims, code, figures, tables, datasets, and experimental results.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Univla: Learning to act anywhere with task-centric latent actions. *arXiv preprint arXiv:2505.06111*, 2025.

Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Guo, Si Xu, Hao Lin, Hongzhi Zang, Quanlu Zhang, Zhaofei Yu, Guoliang Fan, et al. πrl: Online rl fine-tuning for flow-based vision-language-action models. *arXiv preprint arXiv:2510.25889*, 2025.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. *arXiv preprint arXiv:1806.07366*, 2018.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. In *International Conference on Learning Representations*, 2017.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Sinong Geng, Aldo Pacchiano, Andrey Kolobov, and Ching-An Cheng. Improving offline rl by blending heuristics. *arXiv preprint arXiv:2306.00321*, 2023.

Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

Jiatao Gu, Tianrong Chen, David Berthelot, Huangjie Zheng, Yuyang Wang, Ruixiang Zhang, Laurent Dinh, Miguel Angel Bautista, Joshua Susskind, and Shuangfei Zhai. Starflow: Scaling latent normalizing flows for high-resolution image synthesis. *Advances in Neural Information Processing Systems*, 38:120986–121022, 2026.

Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*, 2023.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Dongchi Huang, Zhirui Fang, Tianle Zhang, Yihang Li, Lin Zhao, and Chunhe Xia. Co-rft: Efficient fine-tuning of vision-language-action models through chunked offline reinforcement learning. *arXiv preprint arXiv:2508.02219*, 2025.

Chia-Yu Hung, Qi Sun, Pengfei Hong, Amir Zadeh, Chuan Li, U Tan, Navonil Majumder, Soujanya Poria, et al. Nora: A small open-sourced generalist vision language action model for embodied tasks. *arXiv preprint arXiv:2504.19854*, 2025.

Bingyi Kang, Xiao Ma, Chao Du, Tianyu Pang, and Shuicheng Yan. Efficient diffusion policies for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36:67195–67212, 2023.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29, 2016.

Ivan Kobyzev, Simon JD Prince, and Marcus A Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE transactions on pattern analysis and machine intelligence*, 43 (11):3964–3979, 2020.

Alexander Kolesnikov, André Susano Pinto, and Michael Tschannen. Jet: A modern transformer-based normalizing flow. *arXiv preprint arXiv:2412.15129*, 2024.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The eleventh international conference on learning representations*, 2022.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Yiyang Lu, Qiao Sun, Xianbang Wang, Zhicheng Jiang, Hanhong Zhao, and Kaiming He. Bidirectional normalizing flow: From data to noise and back. *arXiv preprint arXiv:2512.10953*, 2025.

Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

Emile Mathieu and Maximilian Nickel. Riemannian continuous normalizing flows. *Advances in neural information processing systems*, 33:2503–2515, 2020.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. *Advances in neural information processing systems*, 30, 2017.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

Massimiliano Patacchiola, Aliaksandra Shysheya, Katja Hofmann, and Richard E Turner. Transformer neural autoregressive flows. *arXiv preprint arXiv:2401.01855*, 2024.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*, 2023.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pp. 1530–1538. PMLR, 2015.

Pierre Harvey Richemond, Yunhao Tang, Daniel Guo, Daniele Calandriello, Mohammad Gheshlaghi Azar, Rafael Rafailov, Bernardo Avila Pires, Eugene Tarassov, Lucas Spangher, Will Ellsworth, et al. Offline regularised reinforcement learning for large language models alignment. *arXiv preprint arXiv:2405.19107*, 2024.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, et al. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

Junjie Wen, Yichen Zhu, Jinming Li, Minjie Zhu, Zhibin Tang, Kun Wu, Zhiyuan Xu, Ning Liu, Ran Cheng, Chaomin Shen, et al. Tinyvla: Towards fast, data-efficient vision-language-action models for robotic manipulation. *IEEE Robotics and Automation Letters*, 2025.

Chenyu Yang, Denis Tarasov, Davide Liconti, Romain Guntz, Hehui Zheng, and Robert K Katzschmann. Sernf: Sample-efficient real-world dexterous policy fine-tuning via action-chunked critics and normalizing flows. *arXiv preprint arXiv:2602.09580*, 2026.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pp. 1094–1100. PMLR, 2020.

Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. *arXiv preprint arXiv:2403.03954*, 2024.

Shuangfei Zhai, Ruixiang Zhang, Preetum Nakkiran, David Berthelot, Jiatao Gu, Huangjie Zheng, Tianrong Chen, Miguel Angel Bautista, Navdeep Jaitly, and Joshua M Susskind. Normalizing flows are capable generative models. In *Forty-second International Conference on Machine Learning*, 2025.

Qinyu Zhao, Guangting Zheng, Tao Yang, Rui Zhu, Xingjian Leng, Stephen Gould, and Liang Zheng. Simflow: Simplified and end-to-end training of latent normalizing flows. *arXiv preprint arXiv:2512.04084*, 2025.

Guangting Zheng, Qinyu Zhao, Tao Yang, Fei Xiao, Zhijie Lin, Jie Wu, Jiajun Deng, Yanyong Zhang, and Rui Zhu. Farmer: Flow autoregressive transformer over pixels. *arXiv preprint arXiv:2510.23588*, 2025.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pp. 2165–2183. PMLR, 2023.

A Implementation Details

A.1 Training Pipeline

The training pipeline is summarized in Algorithm 1. We first train the AR-NF policy on offline action chunks through maximum-likelihood imitation learning. NF-IQL post-training then proceeds in two phases: we first freeze the actor and warm up the prefix-level critics and state-value function, and subsequently update the critics, value function, and actor using advantage-weighted likelihood optimization over offline action chunks. Finally, we freeze the post-trained AR-NF policy as the teacher and train a one-step student to approximate its inverse mapping through intermediate-state alignment and action-chunk reconstruction, optionally augmented with teacher trajectories generated from prior-sampled latents. The pipeline outputs the post-trained AR-NF policy and the distilled one-step student.

A.2 Reward Relabeling

To alleviate the credit-assignment difficulty caused by sparse rewards, we adopt trajectory-level reward relabeling inspired by HUBL (Geng et al., 2023). For trajectory n, we define the terminal-aware discount as $\gamma_{n,t} = \gamma(1 - d_{n,t})$, where $d_{n,t}$ denotes the terminal indicator, and compute the Monte Carlo return as

$$G_{n,t} = r_{n,t} + \gamma_{n,t} G_{n,t+1}. \tag{16}$$

Let T_n denote the number of transitions in trajectory n. We define its trajectory score as the mean Monte Carlo return,

$$s_n = \frac{1}{T_n} \sum_{t=0}^{T_n - 1} G_{n,t}, \tag{17}$$

and determine a trajectory-level mixing coefficient according to its empirical rank in the offline dataset:

$$\lambda_n = \alpha \frac{1}{N} \sum_{m=1}^{N} \mathbb{I}[s_m \leq s_n], \tag{18}$$

where α controls the relabeling strength. We then define the relabeled reward and bootstrap discount as

$$\widetilde{r}_{n,t} = r_{n,t} + \gamma_{n,t} \lambda_n G_{n,t+1}, \qquad \widetilde{\gamma}_{n,t} = \gamma_{n,t}(1 - \lambda_n). \tag{19}$$

This formulation adaptively interpolates between the observed Monte Carlo return and value-function bootstrapping, assigning greater reliance on observed future returns to higher-return trajectories. For action chunks, we recursively accumulate the relabeled rewards and discounts to construct the TD target for each action prefix.

B Experimental Details

B.1 Model and Training Configurations

RoMAN-Flow is trained in three stages: imitation learning, NF-IQL post-training, and one-step distillation. The complete optimization procedure is summarized in Algorithm 1; here, we describe the benchmark-specific model and training configurations.

Backbones and observations. For LIBERO, MetaWorld-MT50, and the real-robot experiments, we use SmolVLM-500M-Instruct as the multimodal encoder for visual observations, language instructions, and proprioceptive states. The suite-specific LIBERO policies are initialized from a shared checkpoint pretrained on LIBERO-90 for 100k steps. For RoboMimic MH, we follow the perception setup of SERNF (Yang et al., 2026) for a controlled comparison, replacing the VLM with two separate ImageNet-pretrained ResNet-18 encoders for the third-person and wrist-camera RGB observations.

Algorithm 1 RoMAN-Flow Training and Distillation

Input: Offline dataset $\mathcal{D}$
Output: Post-trained policy $\pi_{\theta^\star}$ and one-step student $g_{\psi^\star}$

Initialize π_θ, $\{Q_{\omega_k}, Q_{\bar{\omega}_k}\}_{k=1}^{M}$, and V_ϕ.
Stage I: Imitation learning
for $n = 1$ to N_{IL} **do**
  Sample $(c_t, \mathbf{a}_t) \sim \mathcal{D}$.
  Update θ with $\mathcal{L}_{\mathrm{IL}} = -\mathbb{E}[\log \pi_\theta(\mathbf{a}_t \mid c_t)]$.
end for
Stage II: NF-IQL post-training
Freeze π_θ during value warm-up.
for $n = 1, \ldots, N_{\mathrm{warm}}$ **do**
  Sample $\mathcal{B} \sim \mathcal{D}$ and construct prefix targets $y_{t,j}$.
  Update the critics and value function using $\mathcal{L}_Q$ and $\mathcal{L}_V$.
  Polyak-update the target critics.
end for
Unfreeze π_θ.
for $n = 1$ to N_{IQL} **do**
  Update the critics and value function.
  Compute $A_t = \bar{Q}_{\bar{\omega}}(c_t, \mathbf{a}_t) - V_\phi(c_t)$.
  Update θ using $\mathcal{L}_\pi = -\mathbb{E}[e^{\beta A_t} \log \pi_\theta(\mathbf{a}_t \mid c_t)]$.
end for
Stage III: One-step distillation
Freeze $\pi_{\theta^\star}$ and initialize g_ψ.
for $n = 1$ to N_{align} **do**
  Sample $(c_t, \mathbf{a}_t) \sim \mathcal{D}$.
  Run the teacher forward on $\mathbf{a}_t$ to obtain $\mathbf{z}_t$ and $\{\mathbf{h}_t^{(l)}\}_{l=0}^{L-1}$.
  Run $g_\psi(\mathbf{z}_t; c_t)$ to obtain $\{\widehat{\mathbf{h}}_t^{(l)}\}_{l=0}^{L-1}$ and $\widehat{\mathbf{a}}_t$
  Sample $\mathbf{z}_t^p \sim p_0$
  Run the teacher inverse under c_t to obtain $\{\mathbf{u}_{t,p}^{(l)}\}_{l=0}^{L-1}$ and $\mathbf{a}_t^p$.
  Run $g_\psi(\mathbf{z}_t^p; c_t)$ to obtain $\{\widehat{\mathbf{u}}_{t,p}^{(l)}\}_{l=0}^{L-1}$ and $\widehat{\mathbf{a}}_t^p$.
  Compute $\mathcal{L}_{\mathrm{data}}$ using $\{\widehat{\mathbf{h}}_t^{(l)}\}_{l=0}^{L-1}$ and $\{\mathbf{h}_t^{(l)}\}_{l=0}^{L-1}$, $\widehat{\mathbf{a}}_t$ and $\mathbf{a}_t$.
  Compute $\mathcal{L}_{\mathrm{prior}}$ using $\{\widehat{\mathbf{u}}_{t,p}^{(l)}\}_{l=0}^{L-1}$ and $\{\mathbf{u}_{t,p}^{(l)}\}_{l=0}^{L-1}$, $\widehat{\mathbf{a}}_t^p$ and $\mathbf{a}_t^p$.
  Update ψ using $\mathcal{L}_{\mathrm{data}} + \lambda_p \mathcal{L}_{\mathrm{prior}}$.
end for
return $\pi_{\theta^\star}$ and $g_{\psi^\star}$.

Model sizes. The SmolVLM-based AR-NF policy used for imitation learning and NF-IQL contains approximately 1.45B inference-time parameters, while the corresponding one-step student contains approximately 0.56B parameters. The ResNet-based AR-NF policy and one-step student used on RoboMimic contain approximately 0.95B and 78.2M parameters, respectively. Table 5 summarizes the model architectures and training hyperparameters.

B.2 Evaluation Protocols

All policies are evaluated from fixed checkpoints without parameter updates. We use episode-level success rate as the primary metric and execute each predicted action chunk completely before replanning.

LIBERO. We follow the official LIBERO protocol (Liu et al., 2023) in terms of task suites, success criteria, and initial states. We evaluate LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, and LIBERO-Long, each containing ten tasks. Each task is evaluated over 50 episodes, resulting in 500 episodes per suite. All policies predict and execute 16-step action chunks.

Table 5: Model configurations and training hyperparameters. Paired entries under RoMAN-Flow (IL / NF-IQL) report IL and NF-IQL settings, respectively, while single entries are shared by both stages. One-Step column reports the default distillation configuration, with its two parameter counts corresponding to the SmolVLM- and ResNet-based students. A dash denotes an inapplicable setting.

	IL / NF-IQL				One-Step
Setting	**LIBERO**	**MT50**	**RoboMimic**	**Real Robot**	**Default**
Model configuration					
Hidden dimension	1152	1152	1152	1152	512
Attention heads	16	16	16	16	8
Flow / ViT blocks	6	6	6	6	7
Inference parameters	1.45B	1.45B	0.95B	1.45B	0.56B / 78.2M
Training configuration					
Training steps	20k / 50k	100k / 60k	30k / 50k	50k / 30k	20k
Global batch size	64 / 64	128 / 128	144 / 144	32 / 32	64
Actor LR	$5\times10^{-5}/2\times10^{-5}$	$1\times10^{-4}/5\times10^{-6}$	$1\times10^{-4}/2\times10^{-4}$	$5\times10^{-5}/1\times10^{-5}$	–
Student LR	–	–	–	–	1×10^{-4}
Critic LR	– / 1×10^{-4}	– / 5×10^{-5}	– / 2×10^{-4}	– / 5×10^{-5}	–
Expectile τ	– / 0.80	– / 0.75	– / 0.75	– / 0.75	–
Advantage temp β	– / 10	– / 45	– / 10	– / 3	–
Advantage clip	– / 100	– / 5	– / 20	– / 20	–
HUBL coefficient	– / 0.20	– / 0	– / 0.15	– / 0	–
Critic warm-up	– / 1k	– / 5k	– / 5k	– / 1k	–
Discount γ	– / 0.995	– / 0.995	– / 0.997	– / 0.997	–
Target update rate	– / 0.005	– / 0.005	– / 0.05	– / 0.005	–
Prior-loss weight	–	–	–	–	1
Prior-sample fraction	–	–	–	–	1
Prior-sampling temp	–	–	–	–	0.5
EMA decay	–	–	–	–	0.9999
Optimizer	Adam	Adam	Adam	Adam	Adam
Weight decay	0	0	0	0	0
Gradient clipping	1.0	1.0	1.0	1.0	1.0

MetaWorld-MT50. Following the evaluation protocol of π_{RL} (Chen et al., 2025), we evaluate each of the 50 tasks over ten episodes using four parallel environments and a maximum episode length of 200 steps. All policies predict and execute 5-step action chunks. We report the unweighted mean success rate across the Easy, Medium, Hard, and Very Hard groups.

RoboMimic MH. We follow the training and evaluation protocol of SERNF (Yang et al., 2026) on Lift, Can, and Square. For each of four training seeds, we evaluate the policy on the same 100 fixed initial states, yielding 400 episodes per task and training stage. Each evaluation uses a single environment, a maximum episode length of 500 steps, and 10-step action chunks.

Real-robot evaluation. For real-robot evaluation, we use a dedicated GPU inference server and a separate robot-control workstation. For each task, all policies are evaluated on 30 initializations. Target objects are initialized within predefined evaluation regions that include locations outside the initial-position distribution represented in the offline demonstrations. The resulting protocol therefore covers both in-distribution and spatial out-of-distribution initializations. Each policy predicts 16-step action chunks.

C Ablation Study

C.1 Effect of AR-NF actor capacity.

We study the scaling behavior of the AR-NF actor on RoboMimic Square-MH. Holding the visual encoder, offline dataset, and training and evaluation protocol fixed, we vary only the width, number of attention heads, and depth of the AR-NF actor. As shown in Table 6, the S, B, and L configurations

Table 6: AR-NF actor capacity ablation on RoboMimic Square-MH. SR denotes success rate (%).

Size	Dim.	Heads	Layer Depth	#Params	SR
S	384	6	$[2, 2, 2, 2, 2, 2]$	33.8M	78.0
B	768	12	$[2, 2, 2, 2, 2, 2]$	134.8M	77.0
L	1024	16	$[2, 2, 2, 2, 2, 14]$	466.2M	78.0
XL	1152	16	$[2, 2, 2, 2, 2, 18]$	685.5M	**85.0**

Table 7: Prefix-value aggregation ablation for NF-IQL on RoboMimic Square-MH. SR denotes success rate (%).

Prefix reduction	Evaluation episodes	SR
Last	100	87.0
Mean	100	**88.0**

span 33.8M to 466.2M parameters but achieve nearly identical success rates of 77%–78%. This suggests that increasing capacity within the small-to-medium scale regime does not yet translate into improved modeling of fine-grained action structure. Once scaled to 685.5M parameters, the XL configuration reaches an 85% success rate. These results indicate that sufficiently large actor capacity can better capture the complex, precision-sensitive action distributions required by Square-MH.

C.2 Effect of prefix-value aggregation.

We study how the aggregation of prefix-level critic estimates affects NF-IQL on RoboMimic Square-MH. We keep the visual encoder, offline dataset, actor architecture, critic architecture, optimization schedule, and evaluation protocol fixed. The default *mean* variant averages the target-critic estimates over all action prefixes and critic ensemble members:

$$\bar{Q}_{\text{mean}}(c_t, \mathbf{a}_t) = \frac{1}{MH} \sum_{m=1}^{M} \sum_{j=0}^{H-1} Q_{\bar{\omega}_m}^{(j)}(c_t, a_{t:t+j}).$$

The *last* variant uses only the final action-prefix estimate:

$$\bar{Q}_{\text{last}}(c_t, \mathbf{a}_t) = \frac{1}{M} \sum_{m=1}^{M} Q_{\bar{\omega}_m}^{(H-1)}(c_t, a_{t:t+H-1}).$$

All prefix-level TD targets and critic losses are retained in both variants; only the reduction used to obtain the scalar chunk-level value is changed.

As shown in Table 7, the last-prefix variant obtains an 87% success rate, compared with 88% for mean aggregation. This result suggests that the mean aggregation has a better performance.

C.3 Effect of the distillation source.

We further examine the contribution of the prior-sampled component in the BiFlow distillation objective. We initialize both students from the same NF-IQL teacher and keep the student architecture, optimizer, training steps, data stream, random seed, and evaluation protocol fixed. The *full* variant combines data-induced alignment with the prior-sampled distillation term. The *data-only* variant removes the prior-sampled term and retains only alignment on actions induced by the offline data.

The results are reported in Table 8. The full objective achieves an 88% success rate, compared with 84% for data-only distillation, corresponding to a 4 percentage-point improvement. Thus, under the matched experimental protocol, including prior-sampled supervision provides a measurable performance benefit on RoboMimic Square-MH. This comparison indicates that the prior-sampled term contributes beyond data-induced alignment alone.

Table 8: Distillation-source ablation on RoboMimic Square-MH. SR denotes success rate (%).

Distillation source	Evaluation episodes	SR
Data-only	100	84.0
Full	100	**88.0**

Table 9: Short-run training efficiency on RoboMimic Square-MH.

Offline RL Method	Step Time (s)	Throughput (samples/s)	Peak Memory (GB)
TD3+BC	7.27	2.20	56.3
IQL	0.56	28.6	17.8

C.4 Training efficiency of offline RL updates.

We compare the training cost of TD3+BC and NF-IQL on RoboMimic Square-MH using the same AR-NF initialization, dataset, critic, hardware, and batch size. As shown in Table 9, after warm-up, IQL takes 0.56 seconds per update, compared with 7.27 seconds for TD3+BC, yielding approximately 13$\times$ higher throughput and lower peak GPU memory. This result suggests that conventional sample-based actor-critic updates are less suitable for efficient AR-NF post-training, whereas NF-IQL provides a practical likelihood-based alternative for improving AR-NF policies.

D Benchmarks and Qualitative Examples

D.1 MetaWorld

MetaWorld-MT50 is a public multitask offline manipulation dataset released by LeRobot and built upon the MetaWorld simulation benchmark (Yu et al., 2020). It covers 50 tabletop manipulation tasks performed by a Sawyer robot, spanning button pressing, articulated- object interaction, object grasping and transport, precise assembly, and long-horizon placement. The dataset contains 2,500 expert trajectories and 204,806 transitions, corresponding to 50 trajectories per task. Each sample provides a single-view RGB observation, a 4- dimensional robot state, a 4-dimensional continuous Cartesian action, and a natural-language task description. Following the task partition adopted by π_{RL} (Chen et al., 2025), the 50 tasks are divided into Easy, Medium, Hard, and Very Hard groups containing 28, 11, 6, and 5 tasks, respectively. Figure 4 visualizes representative rollouts from the four difficulty groups: Door Open, Bin Picking, Assembly, and Shelf Place, illustrating articulated-object interaction, cross-bin pick-and-place, precise assembly, and long-horizon elevated placement.

D.2 LIBERO

LIBERO (Liu et al., 2023) is a public language-conditioned manipulation benchmark designed to study knowledge transfer in multitask and lifelong robot learning. The complete benchmark contains 130 tabletop manipulation tasks performed by a Franka Panda robot. Following the common evaluation convention in recent VLA studies, we consider four ten-task suites: LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, and LIBERO-Long, where LIBERO-Long corresponds to LIBERO-10 in the original benchmark. Spatial, Object, and Goal isolate knowledge transfer across spatial layouts, object identities, and language-specified goals, respectively, whereas Long contains longer-horizon tasks involving multiple consecutive manipulation stages. Each task provides 50 human-teleoperated demonstrations, resulting in 2,000 trajectories across the four evaluation suites. RoMAN-Flow uses two RGB views, an 8-dimensional proprioceptive state, and a language instruction. The suite-specific policies are initialized from a shared checkpoint pretrained on the 90 tasks and 4,500 demonstrations of LIBERO-90. Figure 5 presents representative RoMAN-Flow rollouts across the LIBERO suites.

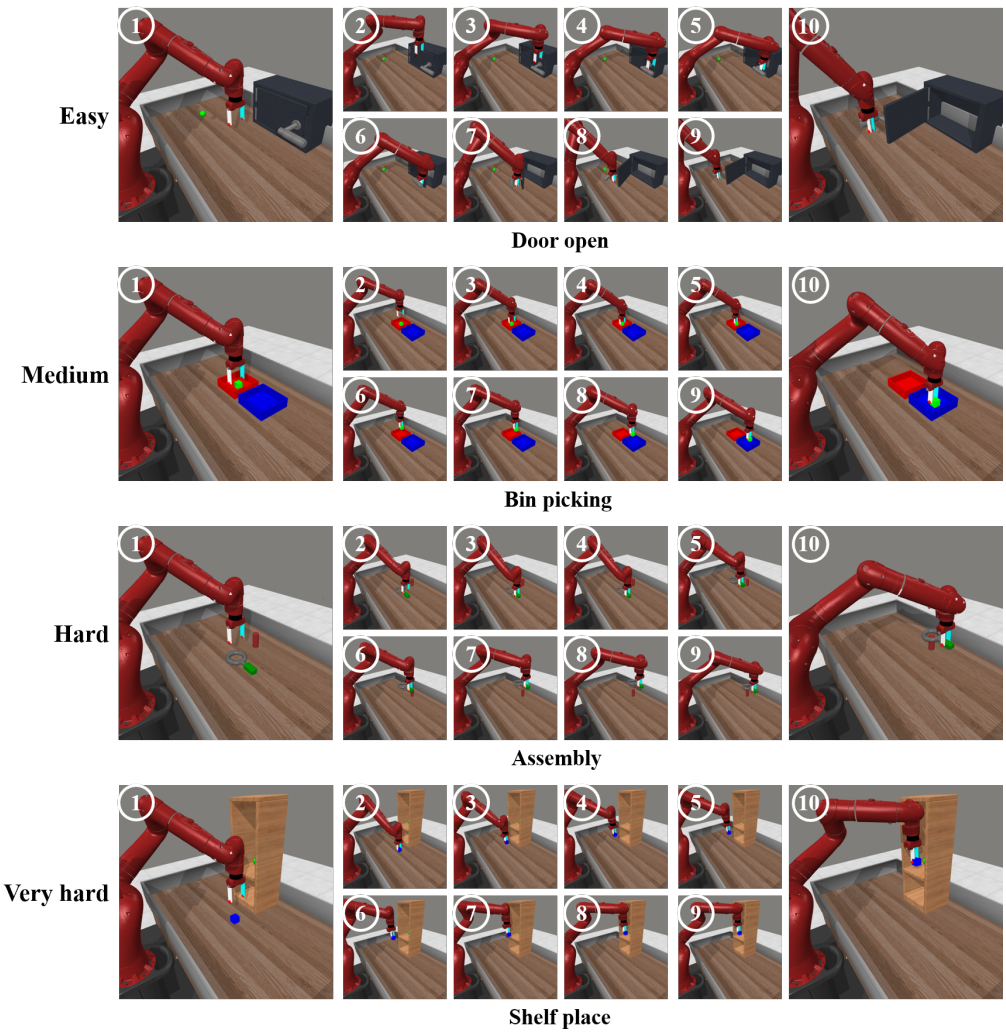


Figure 4: Qualitative results of RoMAN-Flow on representative MetaWorld-MT50 manipulation tasks.

D.3 ROBOMIMIC

RoboMimic (Mandlekar et al., 2021) is a public benchmark for learning robot manipulation policies from offline demonstrations. Its Multi-Human (MH) datasets contain behaviorally diverse demonstrations collected from six human operators with different levels of proficiency. We consider three tasks: Lift, which requires grasping and lifting a cube; Can, which requires transferring a can into a target bin; and Square, which requires precisely placing a square nut onto its corresponding peg. Each task provides 300 demonstrations, resulting in 900 trajectories and 174,614 transitions across the three tasks. To enable a controlled comparison with SERNF (Yang et al., 2026), RoMAN-Flow does not use a VLM or proprioceptive inputs in these experiments. Instead, it observes two 84×84 RGB views from a third-person camera and a wrist-mounted camera and predicts 7-dimensional continuous control commands. The three tasks impose progressively greater manipulation demands, with Square requiring a longer sequence of precise grasping, alignment, and insertion behaviors. Figure 6 presents representative RoMAN-Flow rollouts on Lift, Can, and Square.

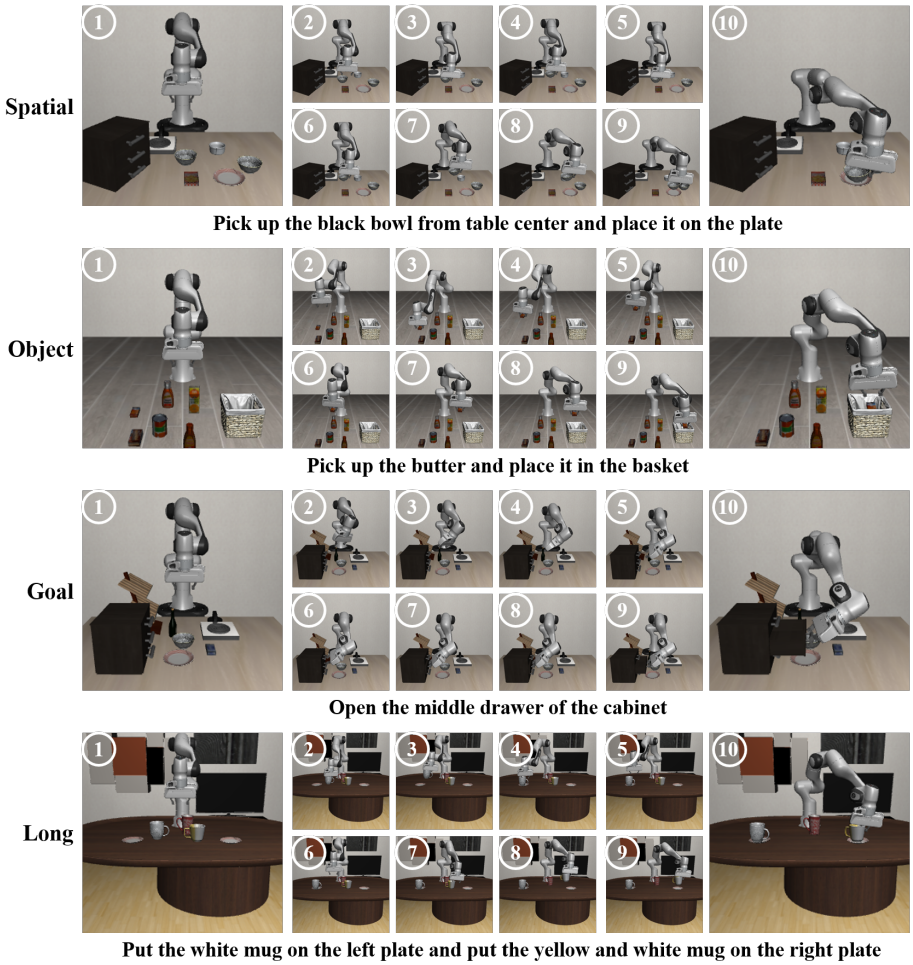


Figure 5: Qualitative rollouts of RoMAN-Flow on representative tasks from the LIBERO benchmark.

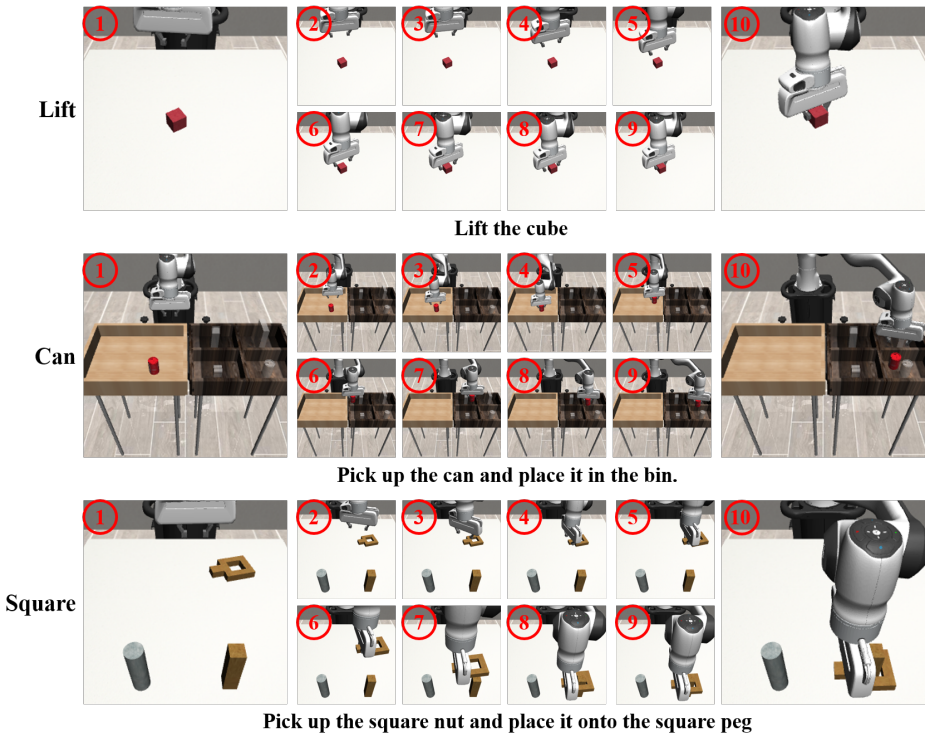


Figure 6: Qualitative rollouts of RoMAN-Flow on the Lift, Can, and Square tasks from RoboMimic MH.