HybridFlow: A 2-NFE Generative Policy for Real-Time Robotic Manipulation

Anonymous Author(s)

Abstract— **Generative policies for robotic manipulation must balance action accuracy with inference latency. We present HybridFlow, a three-stage policy inference procedure requiring two network function evaluations (2-NFE). A Global Jump uses the MeanFlow average velocity to generate a coarse action trajectory; a parameter-free ReNoise interpolation constructs a state at a nonzero refinement time; and a Local Refine queries the instantaneous-velocity limit of the same network at that time. This construction reuses a unified model without distillation. Our analysis characterizes interval-composition errors and the attenuation of endpoint error under ReNoise interpolation. Controlled RoboMimic ablations support the MeanFlow proposal and intermediate-state construction, achieving 95% average success with reused noise and 95.5% with fresh noise, versus 78% for one-step MeanFlow. Across five real-robot settings with all policies running on the same Jetson AGX Thor, HybridFlow improves normalized task performance over 16-step Diffusion Policy by 13–68 points with approximately eightfold lower action-generation latency. Additional experiments demonstrate its compatibility as an action expert in a vision-language-action framework.**

I. Introduction

Deploying generative policies on physical robots requires both accurate actions and timely responses. Diffusion Policy [1], [2] generates action sequences through iterative denoising, typically using 8–32 evaluations. Flow-based policies [3]–[5] provide an alternative transport formulation, but commonly retain several inference steps. In reactive or dynamic manipulation, the scene can change substantially while a policy computes its next action chunk.

MeanFlow [6], [7] offers a different route: instead of integrating an instantaneous velocity field over many short intervals, it learns the average velocity over an interval and can traverse the entire noise-to-action path in one evaluation. This makes it attractive for low-latency control. However, a fast endpoint prediction may lack the precision needed for grasping or contact, and dividing an approximate MeanFlow map into more intervals does not necessarily improve its output.

Our experiments examine how a second network evaluation can improve an already useful one-step policy. In the controlled RoboMimic ablation, one-step MeanFlow achieves 78% average success and HybridFlow reaches 95%. Uniform MeanFlow iteration gives 78%, 72%, and 60% at 2, 4, and 16 NFE. These results motivate using the second evaluation for complementary correction. Physical experiments assess the resulting policy under reactive target interception and dynamic grasping.

We address the inference problem using a property already present in the MeanFlow model: its average velocity becomes the instantaneous velocity when the two conditioning times coincide. HybridFlow assigns two complementary roles to one network. A *Global Jump* predicts a coarse endpoint using the average velocity; *ReNoise* interpolates this endpoint with Gaussian noise; and a *Local Refine* queries the diagonal instantaneous field at the matching intermediate time. ReNoise requires no network evaluation, leaving a total budget of 2-NFE. Both reused and independently redrawn noise are effective in our controlled comparison.

Our contributions are a unified 2-NFE inference procedure, an analysis of its refinement inputs, and controlled evaluation of its components. The analysis relates interval-prediction errors to generated queries and characterizes ReNoise for either noise source. Ablations examine proposal choice, intermediate-state construction, and noise-source choice. Simulation, five physical evaluation settings, and VLA action-expert experiments assess the resulting balance of accuracy and latency.

II. Related Work

Generative manipulation policies. Diffusion Policy [1] models conditional action sequences by denoising. Extensions use 3D representations [2], hierarchical generation [8], and visual-tactile observations [9]. Flow matching [10] instead learns a transport field, with Rectified Flow [11] using straight interpolation paths. MeanFlow learns interval-average velocities [6]; applications and extensions include images [12], [13], PDEs [14], and audio [15]. Learning instantaneous and average fields can interact: Kim et al. [16] find that large-gap average-velocity learning depends on an established instantaneous field and can interfere with it. This motivates examining both training and inference before attributing poor control to a fundamental limitation.

MeanFlow policies. MP1 [17] combines point-cloud conditioning, classifier-free guidance, and dispersive regularization. DM1 [18] also regularizes representations and reports vanilla MeanFlow simulation baselines. OMP [19] adds directional alignment and evaluates ablations removing its added losses. MeanFlow-based One-Step VLA [20] reports successful one-step physical manipulation alongside sensitivity to training hyperparameters and action-chunk length. These studies establish successful applications under their respective observation, training, and control settings. Our evaluation examines a complementary inference design: using a second query to refine an interval-average proposal within one model. We assess this design through controlled component ablations and UMI physical tasks, with performance interpreted relative to the evaluated configurations.

Few-step and refinement-based policies. Consistency Policy [21], ManiCM [22], and OneDP [23] accelerate generation

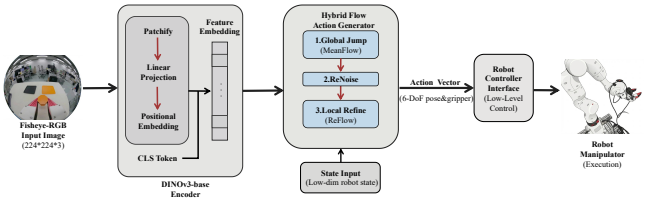

Fig. 1. **Real-robot system architecture.** Fisheye RGB observations are encoded by a pretrained vision encoder, fine-tuned with the policy. Visual features and low-dimensional robot state condition the HybridFlow generator, which produces an action trajectory for the controller.

through distillation or consistency learning. ShortCut [24] conditions on step size; OFP [25] uses self-distillation; and DMPO [26] extends MeanFlow with policy optimization. Energy-based single-pass prediction provides another direction [27]. These approaches modify the training procedure to support fewer inference steps.

RMFlow [13] augments MeanFlow training with a likelihood objective and injects independent noise into its generated output, retaining 1-NFE. HybridFlow instead constructs an intermediate state and evaluates the same network again for instantaneous-velocity correction. STEP [28] uses an auxiliary spatiotemporal predictor to initialize diffusion refinement and velocity-aware perturbation to mitigate execution stalls. HybridFlow obtains its proposal from the MeanFlow network itself. MIP [29] trains a two-step regressor with intermediate supervision and training-time noise, then performs deterministic inference. HybridFlow retains MeanFlow training and combines interval-average and diagonal velocity queries during inference. These methods share the idea of refinement, but differ in their training objectives, proposal construction, and use of additional network evaluations.

III. Method

A. A Unified Average- and Instantaneous-Velocity Model

We learn a visuomotor policy $p(x \mid c)$ over action chunks $x \in \mathbb{R}^d$, where $c = f_\phi(o)$ encodes observations. Rectified Flow transports Gaussian noise at $t = 1$ toward actions at $t = 0$ through an instantaneous field $v(z_t, t, c)$. MeanFlow defines the interval-average velocity

$$u(z_t, r, t, c) = \frac{1}{t-r} \int_r^t v(z_\tau, \tau, c)\, d\tau, \tag{1}$$

and the MeanFlow identity gives

$$u(z_t, r, t, c) = v(z_t, t, c) - (t-r)\frac{d}{dt} u(z_t, r, t, c). \tag{2}$$

At $r = t$, the interval-average field reduces to the instantaneous field: $u(z_t, t, t, c) = v(z_t, t, c)$. Thus, one network u_θ can support long-interval transport and local velocity queries.

Training follows the MeanFlow formulation with (r, t) conditioning and a Jacobian-vector product for the total derivative in Eq. (2). Half of each batch uses $r = t$, where the derivative term vanishes and the objective reduces to instantaneous-velocity learning; the remaining half uses distinct times to learn interval-average transport. The same model is used throughout inference, with no separate refinement network or distillation stage. Figure 1 shows the observation-to-control pipeline.

B. Interval Composition and Query Shift

For $t_k = 1 - k/K$ and $\Delta t = 1/K$, a uniform K-step sampler applies

$$z^{(k+1)} = z^{(k)} - \Delta t\, u_\theta(z^{(k)}, t_{k+1}, t_k, c). \tag{3}$$

Exact transport maps compose consistently across intervals. MeanFlow training on interpolated queries (z_t, r, t) does not explicitly enforce this consistency on the model-generated states encountered during iterative inference.

We refer to the resulting change in state marginals and their coupling with time endpoints as *joint-query distribution shift*. For a fixed query, writing $\widehat{u} = u + \delta$ yields $\widehat{z}_r = z_r - (t-r)\delta$. Prediction error therefore perturbs the next query even under unguided inference. Iterative accuracy depends on both the interval approximation and the states encountered during composition.

Let $z_*^{(k)}$ follow the exact interval maps on the same schedule and from the same initial noise as $z^{(k)}$. Set $E_k = \|z^{(k)} - z_*^{(k)}\|$ and $\Delta t = 1/K$. For the generated query $q_k = (z^{(k)}, t_{k+1}, t_k, c)$, define the approximation error by

$$e_k = u_\theta(q_k) - u^*(q_k). \tag{4}$$

Assume u^* is L-Lipschitz in state along the compared trajectories [30]. Adding and subtracting its value at the generated state, then applying the triangle inequality, gives

$$E_{k+1} \le (1 + L\Delta t)E_k + \Delta t\|e_k\|. \tag{5}$$

Since $E_0 = 0$, unrolling the recursion yields

$$E_K \le \Delta t \sum_{k=0}^{K-1} (1 + L\Delta t)^{K-1-k} \|e_k\|. \tag{6}$$

For $L > 0$ and a uniform error bound $\|e_k\| \le \bar{e}$, this further implies

$$E_K \le \frac{\bar{e}}{L}\left[(1 + L/K)^K - 1\right]. \tag{7}$$

As $K \to \infty$, the bound approaches $\bar{e}(e^L - 1)/L$; for $L = 0$, it reduces to $E_K \le \bar{e}$. This is fixed-interval amplification, not exponential growth with NFE. Local sensitivity and query shift can affect prediction errors [31], [32]. The bound is general; Table II measures the NFE dependence of the evaluated MeanFlow model.

Our RoboMimic experiments measure this dependence through task success (Section IV-A). A related image-generation study reports CIFAR-10 FID 2.80/2.84/5.14/4.34 at 1/2/4/8 MeanFlow steps [33]. These observations motivate a second query designed for local correction.

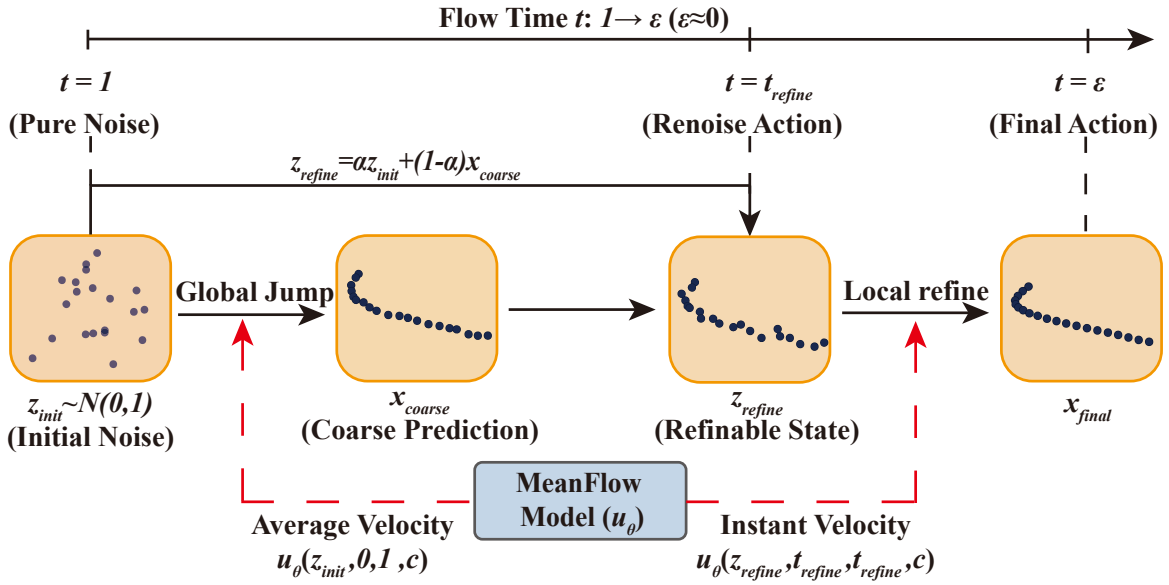

Fig. 2. **HybridFlow: three stages, two network evaluations.** Global Jump predicts an endpoint at $t = 0$; ReNoise returns to $t_{\text{refine}} = \alpha$; Local Refine advances to $t = \epsilon \approx 0$. The stages show computation order, with reused noise as the default. Only the first and third stages evaluate the network.

C. Global Jump, ReNoise, and Local Refine

HybridFlow uses the two query types for distinct purposes (Fig. 2). We write $z_1 = z_{\text{init}}$, $x_c = x_{\text{coarse}}$, and $z_\alpha = z_{\text{refine}}$, with $t_{\text{refine}} = \alpha$. The final query advances to the near-zero terminal time ϵ shown in the figure; $\epsilon = 0$ recovers the ideal data endpoint used in the analysis.

Global Jump. Sample $z_1 \sim \mathcal{N}(0, I)$ and use the interval-average field to predict an endpoint:

$$x_c = z_1 - u_\theta(z_1, 0, 1, c). \tag{8}$$

The interval length is one, so the predicted average velocity represents the full-interval displacement.

ReNoise. Let $\eta = z_1$ for reused noise (default), or let $\eta = \xi$ with $\xi \sim \mathcal{N}(0, I)$ drawn independently of the proposal for the fresh-noise variant. Form

$$z_\alpha = \alpha\eta + (1 - \alpha)x_c, \qquad 0 < \alpha < 1. \tag{9}$$

The query time is set to the interpolation coefficient, $t = \alpha$, giving a time-aligned refinement state without another network evaluation. For a reference endpoint x, use the same draw η to define $z^*_\alpha = \alpha\eta + (1 - \alpha)x$. With $e = x_c - x$,

$$z_\alpha - z^*_\alpha = (1 - \alpha)e. \tag{10}$$

This identity holds for both noise sources: the endpoint error is scaled by $1 - \alpha$ relative to a reference using the same η. It does not require reusing the Global Jump noise or guarantee a match to the training marginal. Noise-source dependence enters the covariance analysis below.

Local Refine. Query the same network on the diagonal $r = t = \alpha$ and take one instantaneous-velocity step to $0 \le \epsilon < \alpha$:

$$\widehat{x} = z_\alpha - (\alpha - \epsilon)u_\theta(z_\alpha, \alpha, \alpha, c). \tag{11}$$

Here $u_\theta(z_\alpha, \alpha, \alpha, c)$ estimates $v(z_\alpha, \alpha, c)$; multiplying by $\alpha - \epsilon$ gives the displacement to the terminal time. The output $\widehat{x} = x_{\text{final}}$ uses two evaluations in total (Algorithm 1).

Algorithm 1 HybridFlow inference (3-stage, 2-NFE)

Require: Model u_θ, encoding c, ratio α, terminal $\epsilon \approx 0$

1: Sample $z_1 \sim \mathcal{N}(0, I)$
2: $x_c \leftarrow z_1 - u_\theta(z_1, 0, 1, c)$ {Global Jump}
3: Set $\eta \leftarrow z_1$ (default) or draw independent $\eta \sim \mathcal{N}(0, I)$
4: $z_\alpha \leftarrow \alpha\eta + (1 - \alpha)x_c$ {ReNoise}
5: $v_\alpha \leftarrow u_\theta(z_\alpha, \alpha, \alpha, c)$ {Local Refine}
6: $\widehat{x} \leftarrow z_\alpha - (\alpha - \epsilon)v_\alpha$
7: **return** $\widehat{x}$

Two-step inference variants. Two-step ReFlow uses two instantaneous-velocity Euler steps. ReFlow–ReNoise–ReFlow replaces HybridFlow's interval-average proposal while retaining its correction pipeline; MeanFlow–ReFlow without ReNoise omits the interpolation. Section IV-B compares these variants.

D. Noise Source and Refinement-Time Alignment

For fixed observation conditioning and any action coordinate, write $V_\eta = \text{Var}(\eta)$, $V_c = \text{Var}(x_c)$, and $C_{\eta c} = \text{Cov}(\eta, x_c)$. Then

$$\text{Var}(z_\alpha) = \alpha^2 V_\eta + (1 - \alpha)^2 V_c + 2\alpha(1 - \alpha)C_{\eta c}. \tag{12}$$

Fresh noise is independent of x_c, giving $C_{\eta c} = 0$. With reused noise, x_c depends on z_1, so the covariance is generally nonzero. For unit noise variance, negligible covariance, and small action variance,

$$\text{Var}(z_\alpha) \approx \alpha^2 V_\eta + (1 - \alpha)^2 V_c \approx \alpha^2. \tag{13}$$

This approximation motivates matching interpolation strength to query time, as in noise-level conditioning [34]; exact marginal matching also depends on the endpoint distribution. Equation (10) requires none of these variance assumptions. The two tested sources give comparable task-level averages (Section IV-B), supporting effective refinement with either noise source in this evaluation. We select α empirically (Fig. 3).

IV. Experiments

We evaluate four RoboMimic tasks [35]—Can, Lift, Square, and Transport—and five physical settings: in-distribution pick-and-place (PP-ID), unseen-color pick-and-place (Pink (OOD)), eyepatch folding (EP), whack-a-mole (WM), and running-car transport (CT). Table I consolidates simulation and real-robot results. We first examine training and inference ablations, then assess task performance and action-expert transfer.

A. Simulation Protocol and MeanFlow Diagnostics

The simulation evaluation uses image-based RoboMimic tasks, a ViT observation encoder, RGB observations from three cameras, and 6-DoF-plus-gripper commands, with 100 episodes per task. Baselines include DDIM [1], ReFlow [11], ShortCut [24], OFP [25], OneDP [23], DMPO [26], and STEP [28]. These cover iterative diffusion/flow, few-step training, and warm-start refinement. Table II additionally evaluates plain MeanFlow and the inference variants of the unified model. Implementation settings are collected in Section IV-E.

Training and time sampling. Simulation training uses 2,000 epochs. Both logit-normal and uniform time sampling reach validation loss near 0.005; RoboMimic success is 70% with uniform sampling and 78% with logit-normal sampling (default). Changing the time-sampling distribution therefore does not improve MeanFlow success in this comparison. We use within-method validation loss to monitor training and task success to compare control quality across regression targets. The separate real-robot training-duration study is described in Section IV-C.

B. Controlled Inference Ablations

The controlled comparison uses one unified checkpoint and 100 episodes per task. Here, ReFlow-mode inference denotes diagonal queries of that model. Plain MeanFlow and ReFlow-mode inference use the uniform schedule $1 \to 0.5 \to 0$ for two-step sampling. The ReFlow–ReNoise–ReFlow variant replaces the full-interval MeanFlow proposal with an instantaneous-field endpoint prediction and retains the same interpolation and final correction. The no-ReNoise variant retains the MeanFlow proposal and final ReFlow correction but omits interpolation.

Table II examines proposal choice, intermediate-state construction, and noise source. Fixing the MeanFlow proposal and diagonal correction, adding ReNoise raises success from 24% to 95%. Fixing ReNoise and correction, replacing the ReFlow proposal with MeanFlow raises success from 82% to 95%. These controls support combining the full-interval proposal with a nonzero-time refinement state. The noise-source comparison below tests whether this improvement requires reusing the initial draw. Separately, MeanFlow iteration degrades at larger NFE; the 58%-to-82% comparison changes both interpolation and inference path. The no-ReNoise variant also yields approximately zero success in the real-robot component check.

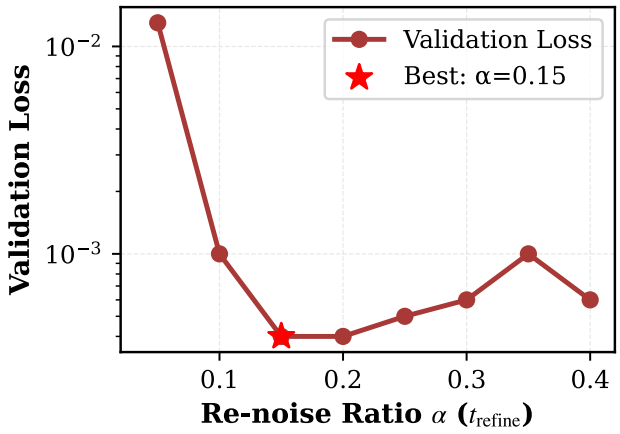

Fig. 3. **ReNoise ratio ablation.** The validation curve favors $\alpha \in [0.15, 0.20]$. We select $\alpha = 0.15$ and use it throughout the evaluated tasks.

Noise-source ablation. Keeping the checkpoint, refinement ratio, and diagonal correction fixed, we change only η in Eq. (9) from the initial draw z_1 to independent $\xi \sim \mathcal{N}(0, I)$. Fresh noise achieves 100%, 100%, 96%, and 86% on Can, Lift, Square, and Transport, respectively: a 95.5% average versus 95% with reused noise. Both sources retain the improvement over the 24% no-ReNoise variant. The tested refinement is effective with either source; the evidence supports constructing a time-aligned state before the diagonal query, rather than requiring reuse of the original draw.

ReNoise strength. Figure 3 reports validation loss as a function of α. At $\alpha = 0.05$, the loss is 0.013; the minimum near $\alpha \in [0.15, 0.20]$ is approximately 4×10^{-4}, and larger values above 0.25 degrade quality. We use $\alpha = 0.15$ for all evaluated tasks without per-task retuning. This supports the choice within the tested settings, while other action distributions or backbones may require a different value.

Comparison with other policies. In Table I(a), HybridFlow reaches 100% on Can and Lift at 2-NFE. On Square it achieves 98%, compared with 96% for 2-NFE STEP, 84% for OneDP, and 83% for DMPO. Transport remains challenging: HybridFlow achieves 82%, versus 86% for 2-NFE STEP and 88% for 16-step DDIM. The task-level results place HybridFlow and STEP at comparable performance under the same denoising budget. Their inference costs have different components: STEP includes a separately trained predictor in addition to denoising, while HybridFlow makes two calls to its unified model.

C. Real-Robot Manipulation

Setup and task specification. Physical experiments use a Franka Research 3 arm and gripper. **All real-robot policy**

TABLE I

SIMULATION AND REAL-ROBOT RESULTS. SIMULATION SUCCESS RATE AND NORMALIZED REAL-ROBOT TASK PERFORMANCE (%).

(a) Simulation results on image-based RoboMimic. Success rate (%) at selected NFE budgets. Best at each (task, NFE) in **bold**. HybridFlow uses a fixed 2-NFE.

		Can				Lift				Square				Transport			
Policy	**Steps**	1	2	4	16	1	2	4	16	1	2	4	16	1	2	4	16
DDIM [1]	1–16	84	94	**100**	**100**	**100**	**100**	**100**	98	72	74	80	76	74	78	84	**88**
ReFlow [11]	1–16	58	52	22	52	42	84	90	76	52	48	56	60	58	48	44	70
ShortCut [24]	1–16	54	66	62	16	30	76	92	86	40	52	62	74	50	54	56	64
OFP [25]	1	82	–	–	–	90	–	–	–	78	–	–	–	72	–	–	–
OneDP [23]	1	78	–	–	–	92	–	–	–	84	–	–	–	64	–	–	–
DMPO [26]	1	**100**	–	–	–	**100**	–	–	–	83	–	–	–	**88**	–	–	–
STEP [28]	2–4	–	**100**	**100**	–	–	**100**	**100**	–	–	96	92	–	–	86	**88**	–
Ours	2	–	**100**	–	–	–	**100**	–	–	–	**98**	–	–	–	82	–	–

(b) Real robot results: all methods deployed on the same Jetson AGX Thor. Normalized task performance (%) across 5 settings, with the same observation encoder, action horizon, robot controller, training data, and comparable action-generator backbone. M-Avg. is the mean of the displayed task scores.

Policy	**Steps**	**T_{inf}**	**PP-ID**	**Pink (OOD)**	**EP**	**WM**	**CT**	**M-Avg.**
MeanFlow	1	~10ms	0.0	0.0	0.0	0.0	0.0	0.0
DDIM [1]	2	~19ms	0.0	0.0	2.5	0.0	0.0	0.5
ReFlow [11]	2	~19ms	10.0	7.5	5.0	2.3	0.0	5.0
DDIM [1]	16	~152ms	63.8	47.5	51.3	47.7	0.0	42.1
Ours	**2**	**~19ms**	**86.3**	**68.8**	**66.3**	**60.7**	**68.3**	**70.1**

PP-ID: Pick-and-Place In-Distribution; **Pink (OOD)**: unseen-color pick-and-place only; **EP**: Eyepatch Folding; **WM**: Whack-a-Mole (normalized aggregate score); **CT**: Running Car Transport (HybridFlow: 43/63; each baseline: 0/63). **T_{inf}**: action-generator inference latency on Jetson AGX Thor, excluding camera acquisition, visual encoding, and robot execution.

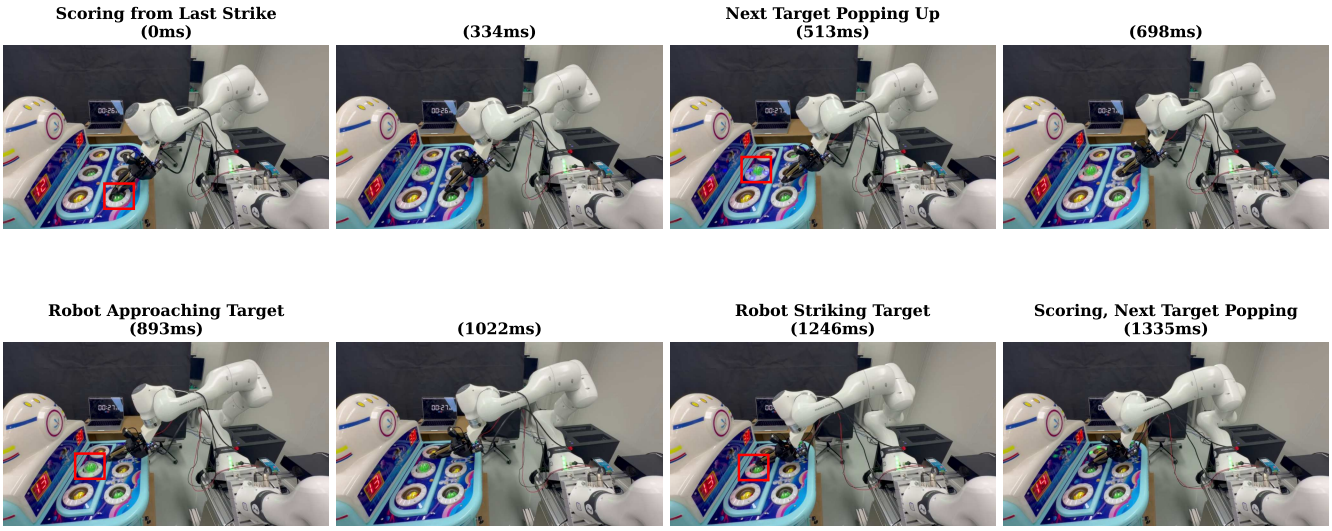

Fig. 4. **Whack-a-mole.** The robot responds to randomly activated targets before they retract. WM uses normalized aggregate score over six rounds (50 points each).

inference, including HybridFlow and every baseline, runs locally on the same Jetson AGX Thor. The UMI setup [36] uses 224×224 fisheye RGB images and low-dimensional robot state (Fig. 1), with a DINOv3-base encoder [37] fine-tuned end-to-end, and approximately 300 demonstrations per task. The baselines share training data, observation encoder, action horizon, controller, and the tuning pipeline, with comparable action-generator backbones. Each method is evaluated over 80 trials per setting for PP-ID, Pink (OOD), and EP, 63 trials for CT, and six rounds for WM. Table I(b) reports each evaluation setting and its unweighted macro-average. Bracketed intervals below are 95% Wilson confidence intervals in percent, assuming independent Bernoulli trials for the binary tasks.

Pick-and-place succeeds when the object is placed in its matching tray. Black and Orange are training/ID colors; Pink (OOD) denotes unseen pink objects; OOD refers only to color. The objects are static, while the arm continues executing

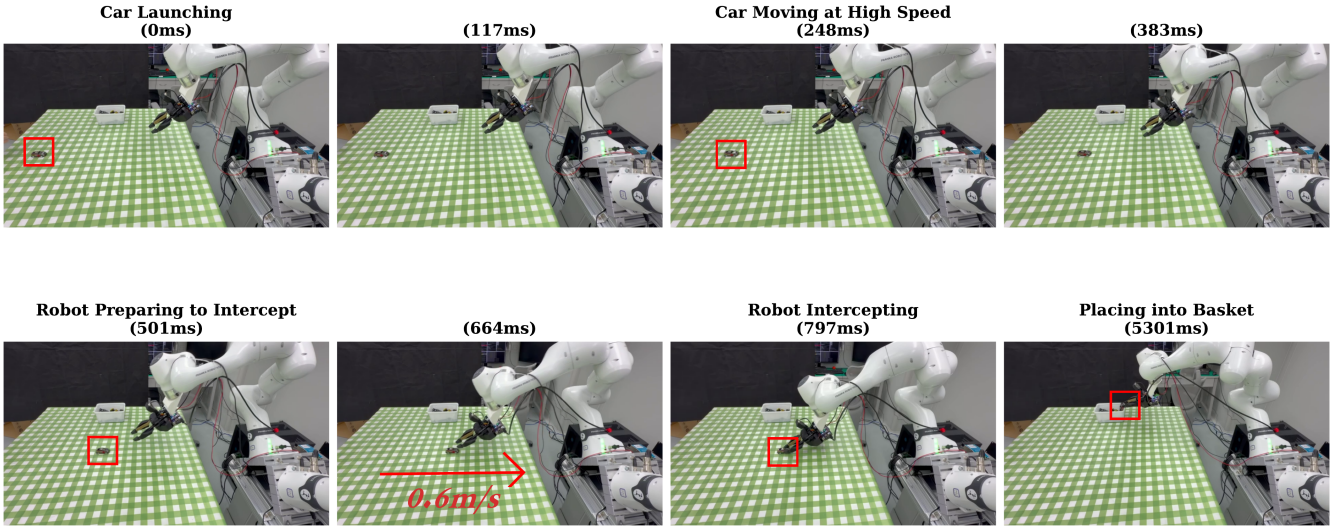

Fig. 5. **Running-car transport.** The robot intercepts a car moving at approximately 0.6 m/s and places it in a basket. HybridFlow succeeds in 43/63 trials (68.3%); each evaluated baseline achieves 0/63.

TABLE II

CONTROLLED ROBOMIMIC INFERENCE ABLATION. AVERAGE SUCCESS (%) OVER THE FOUR TASKS. RN DENOTES RENOISE. ALL VARIANTS QUERY THE SAME UNIFIED CHECKPOINT.

Inference procedure	NFE	Success (%)
Plain MeanFlow	1/2/4/16	78/78/72/60
MeanFlow–ReFlow, no RN	2	24
Two-step ReFlow	2	58
ReFlow–RN–ReFlow	2	82
HybridFlow (same noise)	2	95
HybridFlow (fresh noise)	2	**95.5**

actions between policy predictions. Eyepatch folding requires successive grasps to fold and flatten the fabric. Figures 6 and 7 show the corresponding execution sequences. HybridFlow improves over 16-step DDIM by 15–22.5 percentage points across these three settings.

On PP-ID, Pink (OOD), and EP, respectively, HybridFlow succeeds in 69/80 [77.0, 92.1], 55/80 [57.9, 77.8], and 53/80 [55.4, 75.7] trials. DDIM-16 achieves 51/80 [52.8, 73.4], 38/80 [36.9, 58.3], and 41/80 [40.5, 61.9]; ReFlow-2 achieves 8/80 [5.2, 18.5], 6/80 [3.5, 15.4], and 4/80 [2.0, 12.2]. MeanFlow records 0/80 [0.0, 4.6] in each setting. DDIM-2 records 0/80 [0.0, 4.6] on both pick-and-place settings and 2/80 [0.7, 8.7] on EP.

Reactive control. In whack-a-mole, one of six targets activates at random and retracts after 2 s, with occasional two-target activations outside the demonstration pattern. Each of the six rounds per method has a maximum score of 50 points. Aggregate scores are 0, 0, 7, 143, and 182 out of 300 for MeanFlow, DDIM-2, ReFlow-2, DDIM-16, and HybridFlow, respectively, corresponding to 0%, 0%, 2.3%, 47.7%, and 60.7%. WM is evaluated by normalized aggregate score, not binary task success; its points are not treated as independent trials for confidence intervals.

Dynamic grasping. Running-car transport requires intercepting a toy car moving at approximately 0.6 m/s and placing it in a basket. HybridFlow succeeds in 43/63 trials (68.3%; [56.0, 78.4]), while each evaluated baseline achieves 0/63 (0%; [0.0, 5.7]). At this speed, 152 ms and 19 ms of action generation correspond to approximately 9.1 cm and 1.1 cm of car motion, respectively, quantifying the timing pressure during interception.

Onboard latency and closed-loop response. On the same Thor, DDIM-16 requires 152 ms per action chunk versus 19 ms for HybridFlow. Its sequential denoising thus adds 133 ms before an updated prediction becomes available. The controller continues executing the previously issued trajectory while the next chunk is computed, including in static pick-and-place. Inference therefore overlaps physical execution; the reported evaluation does not pause the arm for each prediction. Longer inference delays the availability of feedback corrections. More denoising steps consume a larger share of the response budget on this edge platform; higher offline generation quality alone does not ensure better closed-loop task success. HybridFlow combines low latency with accurate actions under this shared deployment protocol.

The eightfold speedup measures action-generation latency with observation features already available: camera acquisition, visual encoding, and robot execution are excluded. Auxiliary camera-path measurements are approximately 95 ms on Thor and 62 ms on RTX 5090; all physical rollouts use the Thor path. Training and simulation hardware are listed separately in Table IV.

Execution behavior. Plain MeanFlow fails to generate sufficiently accurate actions for the physical tasks evaluated here. Its predicted actions drove the arm beyond permitted motion limits, causing execution to halt before task completion; ReFlow exhibited the same failure mode. These are action-quality failures despite the policies' low inference latency. Extending real-robot MeanFlow training from the default

TABLE III

VLA ACTION EXPERT TRANSFER. HYBRIDFLOW REPLACES ONLY THE ACTION EXPERT; VLA BACKBONE UNCHANGED.

		LIBERO Benchmark				RoboTwin 2.0	
Method	**Steps**	Spatial	Object	Goal	Avg.	Easy/Hard	Avg.
StarVLA-ReFlow	4	95.6	99.6	96.0	97.1	88.2/88.3	88.3
StarVLA-HybridFlow	2	**96.2**	99.4	**97.5**	**97.7**	**89.6/89.0**	**89.3**

Note: LIBERO Avg. is the unweighted mean of Spatial, Object, and Goal. RoboTwin results are averaged over 3 seeds.

TABLE IV

HYPERPARAMETERS. LEFT: SIMULATION (ROBOMIMIC). RIGHT: REAL ROBOT (UMI).

Simulation (RoboMimic)	
Network	MLP 768-768-768, Mish
Vision	ViT (patch 8, dim 128, 4 heads)
Obs	3 × 96×96 RGB + 9-d state
Action / horizon	7-d / 4 steps
Optimizer	AdamW, lr=10^{-4}, wd=10^{-6}
Batch / epochs	256 / 2000
LR sched	Cosine, warmup 100
EMA	0.995, start epoch 20
Flow ratio	0.5
α	0.15
Train / Infer	RTX 5090

Real Robot (UMI)	
Network	1D UNet [256,512,1024]
Vision	DINOv3-base, fine-tuned
Obs	224×224 fisheye RGB + state
Action / horizon	7-d / 8 steps
Optimizer	AdamW, lr=10^{-4}
Batch / epochs	192 / 200
LR sched	Cosine, warmup 2000 steps
EMA	Power 0.75
Flow ratio	0.5
α	0.15
Train / Infer	RTX 5090 / Jetson AGX Thor

200 to 1,000 epochs yields only a small validation-MSE decrease, from 0.0065 to 0.0061, and does not resolve the poor action quality. This duration study is separate from the 2,000-epoch simulation training. DDIM-2 also exhibited grasp-offset errors.

Plain MeanFlow achieves 78% in the simulation ablation and 0% in the physical evaluation. These results come from separately trained policies with different observations, backbones, and task distributions, rather than direct sim-to-real transfer. The physical failures show that the generated actions lack the accuracy needed for execution in the evaluated UMI settings.

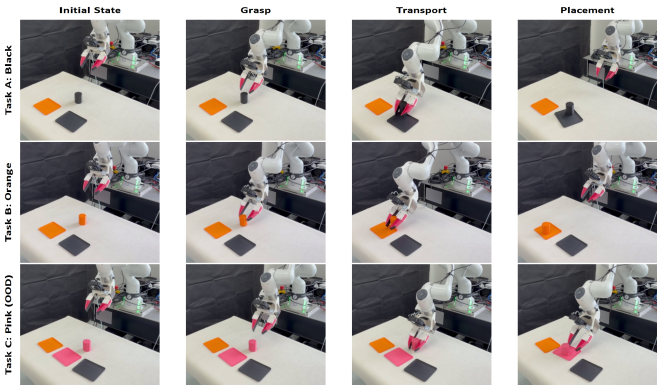

Fig. 6. **Pick-and-place execution.** Each row shows initial state, grasp, transport, and placement. Black and Orange appear in training; Pink (OOD) tests unseen-color generalization only.

D. VLA Action-Expert Transfer

We also integrate HybridFlow as an action expert in StarVLA [38]. The vision-language backbone and connector remain frozen, while the original 4-step ReFlow action expert is replaced with a HybridFlow head using the same training recipe. Table III reports suite-level and aggregate results: on LIBERO, the 2-NFE expert remains competitive across Spatial, Object, and Goal; on RoboTwin 2.0, it reaches 89.3% versus 88.3% for the 4-step baseline, averaged over three seeds. The experiments demonstrate compatibility with the frozen vision-language backbone at half the action-expert evaluation count.

Fig. 7. **Eyepatch folding.** The sequence shows initial state, grasping the left and right edges, and folding and flattening the fabric.

E. Implementation Settings

Table IV summarizes training and network settings. Both calls in Algorithm 1 reuse the same observation encoding; ReNoise is a tensor interpolation.

V. LIMITATIONS

HybridFlow is evaluated on the reported models and tasks; its empirically selected refinement ratio may require retuning for other action distributions. The propagation bound does not establish a causal link between query shift and failure. Plain MeanFlow's action-quality failures are documented in the tested UMI setting; their generality and response to other optimization or representation choices remain open. We do not separately quantify the contributions of action accuracy and feedback delay.

VI. Conclusion

HybridFlow combines a full-interval MeanFlow proposal, time-aligned ReNoise, and diagonal instantaneous-velocity correction in a unified 2-NFE model. Controlled ablations support the proposal and refinement-state construction with either reused or fresh noise. With all real-robot policies on the same Jetson AGX Thor, it improves normalized task performance over DDIM-16 by 13–68 points with approximately eightfold lower action-generation latency, reaching 43/63 successes (68.3%) in dynamic catching. Simulation and VLA experiments demonstrate broader applicability.

References

[1] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *The International Journal of Robotics Research*, 2024.

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

[3] P. I. Team, "π_0: A vision-language-action flow model for general robot control," *arXiv preprint arXiv:2410.24164*, 2024.

[4] ——, "$\pi_{0.5}$: a vision-language-action model with open-world generalization," *arXiv preprint arXiv:2504.16054*, 2025.

[5] S. Liu, B. Li, K. Ma, L. Wu, H. Tan, X. Ouyang, H. Su, and J. Zhu, "RDT2: Exploring the scaling limit of umi data towards zero-shot cross-embodiment generalization," *arXiv preprint arXiv:2602.03310*, 2026. [Online]. Available: https://arxiv.org/abs/2602.03310

[6] Z. Geng, M. Deng, X. Bai, J. Z. Kolter, and K. He, "Mean flows for one-step generative modeling," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[7] Z. Geng, Y. Lu, Z. Wu, E. Shechtman, J. Z. Kolter, and K. He, "Improved mean flows: On the challenges of fastforward generative models," *arXiv preprint arXiv:2512.02012*, 2025. [Online]. Available: https://arxiv.org/abs/2512.02012

[8] Y. Lu, Y. Tian, Z. Yuan, X. Wang, P. Hua, Z. Xue, and H. Xu, "H^3DP: Triply-hierarchical diffusion policy for visuomotor learning," *arXiv preprint arXiv:2505.07819*, 2025.

[9] Y. Tian, S. Cheng, T. Wei, T. Zhou, Y. Zhang, Z. Liu, Q. Han, Z. Yuan, and H. Xu, "ViTaS: Visual tactile soft fusion contrastive learning for visuomotor learning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2026.

[10] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," in *International Conference on Learning Representations (ICLR)*, 2023.

[11] X. Liu, C. Gong, and Q. Liu, "Flow straight and fast: Learning to generate and transfer data with rectified flow," in *International Conference on Learning Representations (ICLR)*, 2023.

[12] Y. Lu, S. Lu, Q. Sun, H. Zhao, Z. Jiang, X. Wang, T. Li, Z. Geng, and K. He, "One-step latent-free image generation with pixel mean flows," *arXiv preprint arXiv:2601.22158*, 2026. [Online]. Available: https://arxiv.org/abs/2601.22158

[13] Y. Huang, S.-H. Wang, A. L. Bertozzi, and B. Wang, "RMFlow: Refined mean flow by a noise-injection step for multimodal generation," in *International Conference on Learning Representations (ICLR)*, 2026, arXiv:2602.00849.

[14] Z. Shen, Z. Wang, J. Xin, and Z. Zhang, "Two-step diffusion: Fast sampling and reliable prediction for 3d keller–segel and kpp equations in fluid flows," *arXiv preprint arXiv:2601.20024*, 2026.

[15] R. Shimizu, X. Jiang, and N. Mesgarani, "Meanflow-tse: One-step generative target speaker extraction with mean flow," *arXiv preprint arXiv:2512.18572*, 2025.

[16] J.-Y. Kim *et al.*, "Understanding, accelerating, and improving meanflow training," in *CVPR*, 2026.

[17] J. Sheng, Z. Wang, P. Li, and M. Liu, "MP1: MeanFlow tames policy learning in 1-step for robotic manipulation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 40, no. 22, 2026, pp. 18 532–18 539.

[18] G. Zou, H. Wang, H. Wu, Y. Qian, Y. Wang, and W. Li, "Dm1: Meanflow with dispersive regularization for 1-step robotic manipulation," *arXiv preprint arXiv:2510.07865*, 2025. [Online]. Available: https://arxiv.org/abs/2510.07865

[19] H. Fang, Y. Huang, Y. Zhao, P. Weng, X. Li, and Y. Ban, "OMP: One-step MeanFlow policy with directional alignment," in *International Conference on Machine Learning (ICML)*, 2026. [Online]. Available: https://arxiv.org/abs/2512.19347v3

[20] Y. Chen, X. Ma, and B. Zhao, "Mean-Flow based one-step vision-language-action," *arXiv preprint arXiv:2603.01469*, 2026.

[21] A. Prasad, K. Lin, J. Wu, L. Zhou, and J. Bohg, "Consistency policy: Accelerated visuomotor policies via consistency distillation," *arXiv preprint arXiv:2405.07503*, 2024.

[22] G. Lu, Z. Gao, T. Chen, W. Dai, Z. Wang, W. Ding, and Y. Tang, "ManiCM: Real-time 3d diffusion policy via consistency model for robotic manipulation," *arXiv preprint arXiv:2406.01586*, 2025. [Online]. Available: https://arxiv.org/abs/2406.01586

[23] Z. Wang, Z. Li, A. Mandlekar, Z. Xu, J. Tremblay, Y.-W. Chao, S. Birchfield, B. Wen, M. Zhou, M.-Y. Liu, and D. Fox, "One-step diffusion policy: Fast visuomotor policies via diffusion distillation," *arXiv preprint arXiv:2410.21257*, 2024.

[24] K. Frans, D. Hafner, S. Levine, and P. Abbeel, "One step diffusion via shortcut models," *arXiv preprint arXiv:2410.12557*, 2025. [Online]. Available: https://arxiv.org/abs/2410.12557

[25] S. Li, L. Sun, and Y. Chen, "One-step flow policy: Self-distillation for fast visuomotor policies," *arXiv preprint arXiv:2603.12480*, 2026.

[26] G. Zou, H. Wang, H. Wu, Y. Qian, Y. Wang, and W. Li, "One step is enough: Dispersive meanflow policy optimization," *arXiv preprint arXiv:2601.20701*, 2026.

[27] J. Jia, T. Yang, X. Chen, C. Liu, and W. Zhang, "Fast visuomotor policy for robotic manipulation," *arXiv preprint arXiv:2510.12483*, 2025.

[28] J. Li, Y. Cong, Y. Wang, H. Xia, S. Huang, Y. Zhang, N. Xu, and G. Dai, "STEP: Warm-started visuomotor policies with spatiotemporal consistency prediction," *arXiv preprint arXiv:2602.08245*, 2026, ICML 2026.

[29] C. Pan *et al.*, "Much ado about noising: Dispelling the myths of generative robotic control," *arXiv:2512.01809*, 2025.

[30] S. Chen, S. Chewi, H. Lee, Y. Li, J. Lu, and A. Salim, "The probability flow ode is provably fast," *arXiv preprint arXiv:2305.11798*, 2023. [Online]. Available: https://arxiv.org/abs/2305.11798

[31] C. Mooney, Z. Wang, J. Xin, and Y. Yu, "Global well-posedness and convergence analysis of score-based generative models via sharp lipschitz estimates," in *The Thirteenth International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=r3cWq6KKbt

[32] Z. Zhou and W. Liu, "An error analysis of flow matching for deep generative modeling," in *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, ser. Proceedings of Machine Learning Research, vol. 267. PMLR, 2025, pp. 78 903–78 932.

[33] X. Ai, Y. He, A. Gu, R. Salakhutdinov, J. Z. Kolter, N. M. Boffi, and M. Simchowitz, "Joint distillation for fast likelihood evaluation and sampling in flow-based models," in *The Fourteenth International Conference on Learning Representations*, 2026. [Online]. Available: https://openreview.net/forum?id=8uZ5UdIul2

[34] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[35] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, "What matters in learning from offline human demonstrations for robot manipulation," in *Conference on Robot Learning (CoRL)*, 2021.

[36] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song, "Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots," in *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

[37] O. Siméoni, H. V. Vo, M. Seitzer, F. Baldassarre, M. Oquab, C. Jose, V. Khalidov, M. Szafraniec, S. Yi, M. Ramamonjisoa, F. Massa, D. Haziza, L. Wehrstedt, J. Wang, T. Darcet, T. Moutakanni, L. Sentana, C. Roberts, A. Vedaldi, J. Tolan, J. Brandt, C. Couprie, J. Mairal, H. Jégou, P. Labatut, and P. Bojanowski, "Dinov3," *arXiv preprint arXiv:2508.10104*, 2025. [Online]. Available: https://arxiv.org/abs/2508.10104

[38] StarVLA Community, "Starvla: A lego-like codebase for vision-language-action model developing," *arXiv preprint arXiv:2604.05014*, 2026.