UniFlow: A Unified Multi-Modal Policy for Robot Manipulation

Ge Yan[1] **Shun Iwase**[2] **Yuzhi Fan**[1] **Dieter Fox**[1,3] **Sergey Zakharov**[2] **Katherine Liu**[2]
[1]University of Washington [2]Toyota Research Institute [3]Allen Institute for AI

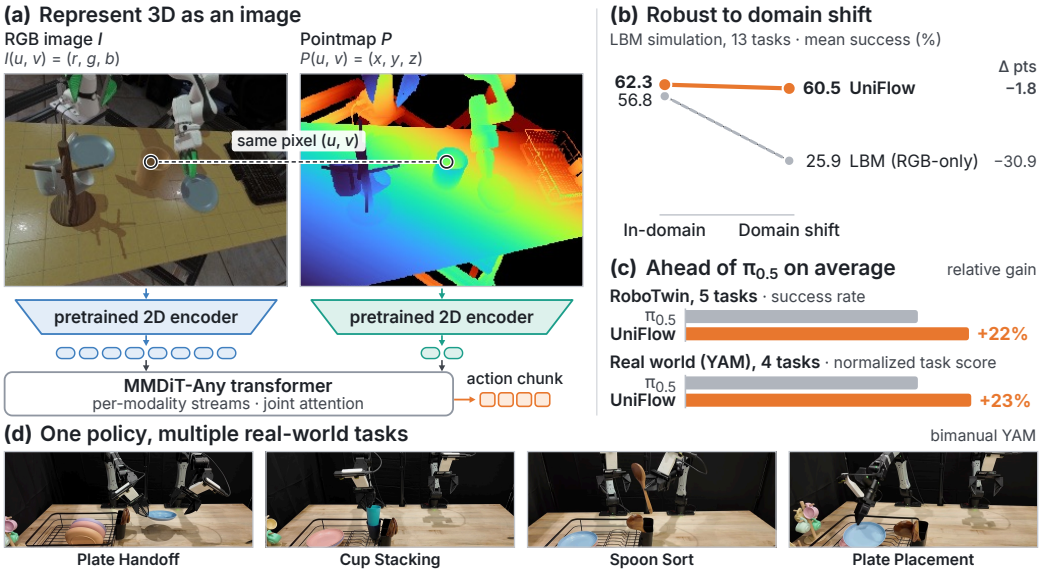


Figure 1: **UniFlow encodes 3D as an image and stays robust under domain shift.** **(a)** A pointmap P stores the 3D point seen at each pixel of the RGB image I, so pretrained 2D encoders read both. **(b)** Over 13 LBM tasks, UniFlow loses 1.8 points under domain shift, the RGB-only LBM policy 30.9. **(c)** Relative gain over $\pi_{0.5}$ on average. **(d)** One UniFlow policy on multiple real-world tasks.

Abstract: Visuomotor manipulation policies face a persistent trade-off: image-only policies inherit the semantic priors of large-scale 2D pretraining but are brittle to lighting, texture, and viewpoint shift, whereas 3D-only policies are robust to such factors yet are often trained from scratch. Some recent multi-modal policies combine both streams but typically process 3D through a from-scratch encoder, preserving this asymmetry. We present **UniFlow**, a unified 2D+3D co-training recipe for manipulation in which 3D observations are represented as image-shaped pointmaps (per-pixel 3D coordinates) so that pretrained 2D backbones can be used to encode both modalities, and RGB and pointmap streams are fused by a unified diffusion transformer with per-modality modulation. On average across thirteen dexterous manipulation tasks from the challenging Large Behavior Model (LBM) simulation benchmark, UniFlow outperforms a strong RGB-only baseline in a distribution-shifted setting. UniFlow also outperforms the generalist vision-language-action (VLA) model $\pi_{0.5}$ when averaged over five RoboTwin bimanual tasks as well as four real-world tasks on the bimanual YAM platform. In ablations, encoding pointmaps with a pretrained 2D backbone outperforms the large-scale pretrained point-cloud encoders we tested. Project website: https://geyan21.github.io/UniFlow/.

Keywords: Visuomotor Policy, Multimodal Representation, Imitation Learning

10th Conference on Robot Learning (CoRL 2026), Austin, Texas, USA.

1 Introduction

Visuomotor imitation learning has emerged as a practical recipe for training robotic manipulation policies from demonstrations, with diffusion- and flow-matching-based action heads pushing in-distribution performance to new levels on cluttered, contact-rich tasks [1, 2]. Yet a persistent gap remains between performance on the training distribution and performance under even modest deployment-time changes in lighting, background texture, or viewpoint [3].

Most contemporary policies commit to a single visual modality. Image-only methods inherit the semantic richness of large-scale 2D pretraining, learning to recognize objects, parse textures, and ground language. Yet they still suffer in unseen environments, e.g., with unseen backgrounds [4]. 3D-only methods, which operate on point clouds or voxelized scenes [5, 6], are robust to such nuisances by leveraging accurate geometry but lack the internet-scale pretraining that gives 2D methods their semantic priors [7]. Recent multi-modal policies [8] combine the two but preserve an architectural asymmetry: a pretrained 2D backbone for the image branch, a from-scratch encoder for the 3D branch, where 3D contributes geometric invariance without the semantic prior.

In this work, we propose UniFlow, a unified multi-modal policy that learns from both 2D and 3D inputs. UniFlow builds on two technical insights. First, by representing 3D observations as *pointmaps*—images whose pixels store the 3D coordinates of the observed scene points instead of colors—we can encode geometry with 2D backbones that have been pre-trained on large-scale image data instead of bespoke 3D point-cloud encoders, so that the 3D stream can also benefit from 2D pretraining. Second, modeling 2D and 3D information as two complementary information streams for policy inference can mitigate the weaknesses of both and lead to more robust policies.

Concretely, UniFlow encodes RGB images and pointmaps with two separately pretrained 2D backbones and fuses them with noisy action tokens in a multi-modal diffusion transformer (MMDiT) [9] with per-stream adaptive layer normalization (AdaLN) [10] and joint attention, trained with the consistency flow-matching objective of ManiFlow [2].

On average over 13 dexterous manipulation tasks from the Large Behavior Model (LBM) benchmark [3], evaluated both in-domain and under domain shift, UniFlow matches the RGB-based LBM policy in-domain (62.3% vs. 56.8%) and substantially exceeds it under shift (60.5% vs. 25.9%), confirming that 2D+3D co-training is effective in reducing the in-distribution / out-of-distribution gap that single-modality policies can struggle with. UniFlow further outperforms the generalist vision-language-action (VLA) model $\pi_{0.5}$ [11] on average over five RoboTwin [12] bimanual tasks (44.4% vs. 36.4% success) and four real-world YAM bimanual tasks (59.2 vs. 48.2 average normalized task score). **Our contributions are as follows:**

- A pointmap-based 3D stream that enables pretrained 2D backbones to encode geometry, so that the 3D stream also benefits from large-scale 2D pretraining; in our ablations, it outperforms the point-cloud encoders we tested, including pretrained ones.
- A multi-modal policy that fuses RGB, pointmap, and action tokens via an MMDiT-style backbone with per-modality AdaLN and joint attention.
- Empirical validation across 18 simulation tasks (13 LBM + 5 RoboTwin) and 4 real-world bimanual YAM tasks, showing that UniFlow generally outperforms single-modality baseline methods under domain shift in simulation and achieves a higher average score than $\pi_{0.5}$ in the real world, with ablations of the 3D encoder, its pretraining, and the fusion backbone.

2 Related Work

Visuomotor imitation learning. Modern visuomotor policies cast action prediction as a conditional generative modeling problem. Diffusion Policy [1] generates action chunks with a denoising diffusion head conditioned on visual observations and has become a standard baseline for image-based manipulation. More recent work [11] has scaled this recipe to larger models and broader task suites,

and has replaced the original denoising diffusion probabilistic model (DDPM) objective with flow matching [13]. UniFlow inherits this generative-action-head perspective and uses the consistency flow-matching objective of ManiFlow [2].

3D representations for robot manipulation. A parallel line of work argues that 3D representations are inherently more robust than 2D images for manipulation, since scene geometry is invariant to lighting, texture, and viewpoint. Point-cloud policies such as DP3 [5] encode the scene with a PointNet-style backbone [14, 15] into a compact feature that conditions a diffusion head; voxel- and neural-radiance-field-based representations [6, 16, 17, 18] take a different geometric route. A common limitation is that these 3D backbones are trained from scratch on a small amount of manipulation data, rather than initialized from the large-scale 2D pretraining that powers modern vision encoders. Pretrained point-cloud encoders [19, 20] offer an alternative, but in our experiments both underperform a pretrained 2D backbone applied to pointmaps (Section 5.4). Recent geometry foundation models [21] have popularized pointmaps, dense per-pixel (x, y, z) maps aligned with an RGB image, as a unifying intermediate for 3D vision. PointMapPolicy [8] adopts pointmaps as policy input but pretrains only the RGB encoder, training the pointmap encoder from scratch and preserving the same asymmetry. UniFlow instead applies a pretrained 2D backbone to both modalities, transferring learned priors to the 3D branch as well.

Multi-modal fusion in policy learning. Multi-modal policies typically fuse modalities by token concatenation [22], cross-attention [2], or feature addition. A recurring failure mode is that a jointly trained network does no better than its best single-modality counterpart [23], e.g., through single-modality collapse: when one modality carries a stronger learning signal, the network leans on it and under-trains the other, becoming brittle when the dominant modality is perturbed at test time. This can be problematic in 2D+3D fusion, where a pretrained RGB backbone may dominate joint training over a less-pretrained 3D branch. Per-modality modulation, as in MMDiT [9], mitigates this: each modality receives its own scale and shift parameters in adaptive layer normalization [10] while the modalities interact through joint attention. UniFlow adopts this design for manipulation, applying per-modality MMDiT modulation to the action, RGB, and pointmap streams, with proprioception, language, the diffusion timestep, and the step size injected as global conditioning. Unlike ManiFlow [2], whose DiT-X backbone conditions an action-only token stream on the observations through cross-attention, UniFlow processes the action, RGB, and pointmap streams jointly.

3 UniFlow

UniFlow is a unified visuomotor policy that consumes multi-view RGB, pointmaps, robot proprioception, and a language instruction, and outputs a chunk of H future actions. Formally, let $o_t = (\{I_t^c\}_{c=1}^C, \{P_t^c\}_{c=1}^C, s_t)$ denote the observation at time t comprising C multi-view RGB images I_t^c, aligned pointmaps P_t^c, and proprioceptive state s_t. Given o_t and a language instruction ℓ, UniFlow learns a policy

$$\pi_\theta(a_{t:t+H} \mid o_t, \ell), \tag{1}$$

that predicts an action chunk $a_{t:t+H} = (a_t, a_{t+1}, \ldots, a_{t+H-1})$ of length H.

The model has three components (Figure 2): a two-stream visual encoder that produces per-camera token sequences for RGB and pointmaps (Section 3.1); a unified MMDiT-Any backbone with per-modality modulation (Section 3.2); and a consistency flow-matching action head trained jointly with the rest of the network (Section 3.3). At inference, an action chunk is generated from Gaussian noise (App. C.7). Background on flow matching, AdaLN, MMDiT, and the two pretrained backbones is summarized in App. C.8.

3.1 Encoding Multi-modal Observations

UniFlow uses two independent pretrained 2D image backbones, one per modality, to encode RGB and pointmap observations; both are initialized from public pretrained weights and fine-tuned jointly during policy learning. We use separate backbones because RGB and pointmaps carry different

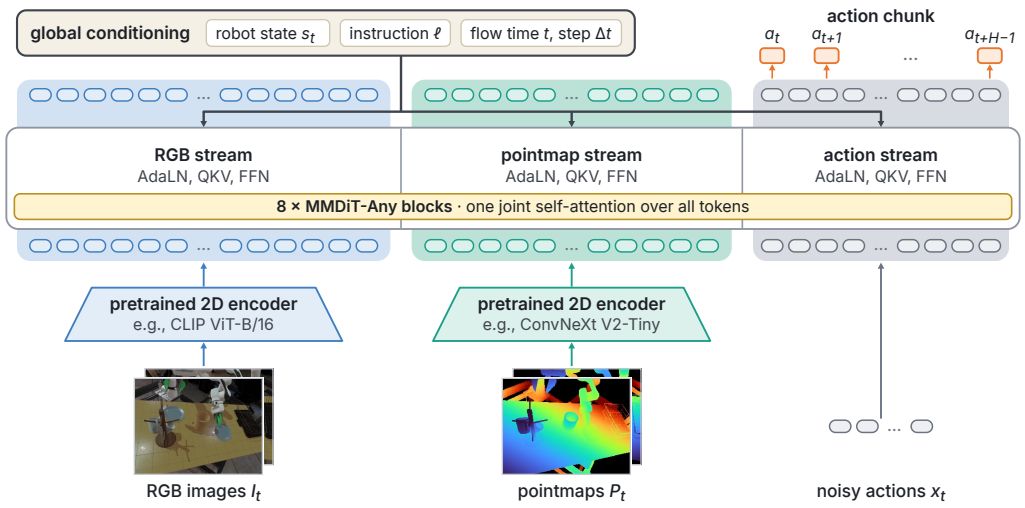


Figure 2: **UniFlow architecture.** RGB images I_t and their pixel-aligned pointmaps P_t are encoded by pretrained 2D encoders, fine-tuned end to end, into RGB and pointmap tokens. Together with the noisy action tokens x_t, they pass through 8 MMDiT-Any blocks: each stream keeps its own AdaLN modulation, attention projections and feed-forward network, and one joint self-attention spans all tokens. A global conditioning vector (robot state, instruction, flow time t and step size Δt) modulates every stream.

information (semantics vs. geometry), which a single shared encoder would have to model jointly. Representing 3D as image-shaped pointmaps allows pretrained 2D backbones to encode geometry, avoids the subsampling and neighborhood operations (e.g., farthest point sampling and k-nearest-neighbor grouping) required by point-cloud architectures, and keeps the 3D stream more spatially aligned with the RGB stream for cross-modal fusion.

RGB encoder. Each 224×224 camera image is normalized with the CLIP mean and standard deviation and encoded by a pretrained CLIP ViT-B/16 [24], a Vision Transformer trained with contrastive image–text pretraining. The ViT splits the image into a 14×14 grid of non-overlapping 16×16 patches and outputs one 768-dimensional token per patch, i.e., 196 spatial tokens per camera. At training time, we apply color jitter to all RGB views and a small random crop, resize, and in-plane rotation to non-wrist RGB views.

Pointmap encoder. For each camera, we construct a *pointmap*: a 3-channel image at the same 224×224 resolution as the RGB image, where the channels at pixel (u, v) store the 3D coordinates (x, y, z) of the scene point projecting to that pixel. Pointmaps are computed from registered depth z and the camera intrinsics by back-projection, $x = (u - c_x)\, z/f_x$ and $y = (v - c_y)\, z/f_y$, where (f_x, f_y) are the focal lengths and (c_x, c_y) the principal point; this requires depth registered to the RGB camera (e.g., from a depth sensor or stereo) and known intrinsics. We keep each pointmap in its own camera's frame rather than a shared world frame. Pointmaps are clipped to a per-camera bounding box, normalized to $[-1, 1]$ per channel, and encoded by a pretrained ConvNeXt V2 Tiny [25], a convolutional network co-designed with masked-autoencoder pretraining, producing a 7×7 feature map of dimension 768, i.e., 49 tokens per camera. No augmentation is applied to pointmaps to preserve the physical coordinates encoded in their channels.

Global conditioning. Robot proprioception (end-effector pose and gripper state) is projected by a small adaptor MLP into an embedding. When language conditioning is enabled, the instruction is encoded by a pretrained CLIP text encoder [24] and its pooled embedding is concatenated with the proprioception embedding. This vector, together with the diffusion timestep and the consistency step size Δt (Section 3.3), forms the global conditioning c that drives per-modality AdaLN in the

MMDiT-Any backbone. Here, *global* means that c modulates every token of a stream instead of being tokenized; it is recomputed at every policy query.

3.2 Multi-Modal Fusion

For an observation with N_{cam} cameras, the RGB stream produces $N_{\text{cam}} \cdot L_{\text{rgb}}$ tokens of dimension D_{rgb}, and the pointmap stream produces $N_{\text{cam}} \cdot L_{\text{pm}}$ tokens of dimension D_{pm}, where $L_{\text{rgb}} = 14 \times 14 = 196$ and $L_{\text{pm}} = 7 \times 7 = 49$ are the per-camera spatial token counts of the two backbones; the pointmap stream thus adds only a quarter as many tokens as the RGB stream. Rather than summing or averaging the two streams, UniFlow maintains RGB and pointmap streams separate up to the transformer backbone, and it keeps per-patch tokens rather than one pooled (CLS) embedding per camera as in LBM [3].

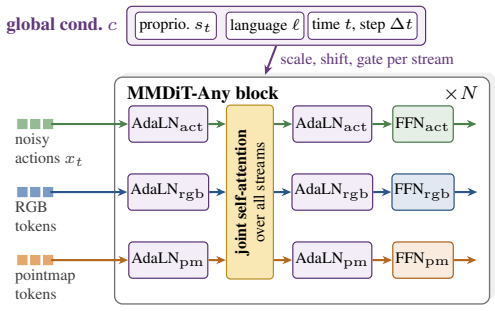


Figure 3: **An MMDiT-Any block:** per-modality *parallel modulation* around a joint self-attention.

To fuse the multi-modal inputs, we build the action transformer from MMDiT [9] blocks generalized to admit an arbitrary number of token streams, which we call *MMDiT-Any* (Figures 2 and 3); the noisy action tokens, the RGB tokens, and the pointmap tokens each form a stream. Each block uses a single *joint self-attention* over the concatenation of all streams, so that each stream can attend to the others within one global attention map, while each stream is modulated in parallel by its own AdaLN scale/shift [10]. Concretely, for an input token x_m of stream m and global conditioning c, each block applies

$$x_m \leftarrow x_m + \alpha_m^{\text{attn}}(c) \cdot \text{Attn}\big(\{\text{AdaLN}_{m'}(x_{m'}, c)\}_{m'}\big)_m, \tag{2}$$

$$x_m \leftarrow x_m + \alpha_m^{\text{ffn}}(c) \cdot \text{FFN}_m\big(\text{AdaLN}_m(x_m, c)\big), \tag{3}$$

where $\text{AdaLN}_m(x, c)$ layer-normalizes x and then scales and shifts it with parameters regressed from c, $\alpha_m(c)$ are learned gates, and $\text{Attn}(\cdot)_m$ is the joint-attention output at the positions of stream m; α_m, AdaLN_m, FFN_m, and the query, key, and value projections inside Attn are all stream-specific.

During training, we randomly drop modalities to encourage robustness. For each sample, each modality $m \in \{\text{rgb}, \text{pointmap}\}$, and each camera, we sample a Bernoulli mask at rate $p_m = 0.1$; masked tokens are zeroed and excluded from the joint self-attention.

3.3 Training Objective: Consistency Flow Matching

We adopt the consistency flow matching objective of ManiFlow [2] as our training loss. Given a ground-truth action chunk x_1 and noise $x_0 \sim \mathcal{N}(0, \mathbf{I})$, the noisy sample $x_t = (1-t)\,x_0 + t\,x_1$ has target velocity $x_1 - x_0$; the velocity network $v_\theta(x_t, t, \Delta t \mid o, \ell)$ is conditioned on the observation o and instruction ℓ (through the token streams and the global conditioning) and on a step size Δt, so that it predicts the average velocity over a finite interval [26]. We jointly minimize a flow matching term (with $\Delta t = 0$) and a consistency term that constrains v_θ to match the average velocity from x_t to an endpoint extrapolated by an exponential-moving-average (EMA) teacher θ^-:

$$\mathcal{L}(\theta) \;=\; \mathbb{E}\Big[\big\|v_\theta(x_t, t, 0 \mid o, \ell) - (x_1 - x_0)\big\|^2 \;+\; \big\|v_\theta(x_t, t, \Delta t \mid o, \ell) - \tilde{v}_{\text{tgt}}\big\|^2\Big], \tag{4}$$

where t is drawn from a Beta distribution for the flow term and $t, \Delta t \sim \mathcal{U}[0, 1]$ for the consistency term, sampled continuously to avoid the bias of discrete-time consistency training. We refer readers to ManiFlow for the construction of $\tilde{v}_{\text{tgt}}$. The two terms are mixed within each mini-batch with a fixed ratio (flow:consistency $= 0.75{:}0.25$ in our experiments).

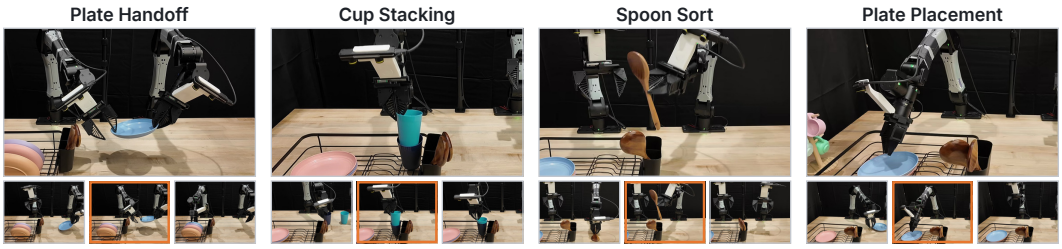


Figure 4: **Real-world tasks.** Rollouts of one multi-task UniFlow policy on the bimanual YAM.

4 Experimental Setup

We evaluate UniFlow in three settings: (i) the **LBM** benchmark of 13 dexterous manipulation tasks, with matched in-domain and domain-shift conditions, where the main comparison is against an RGB-only LBM policy [3]; (ii) the **RoboTwin** simulator, with a 5-task multi-task evaluation against the generalist VLA $\pi_{0.5}$ [11] and the 3D policy ManiFlow [2]; and (iii) four **real-world** bimanual tasks on the YAM platform, again against $\pi_{0.5}$.

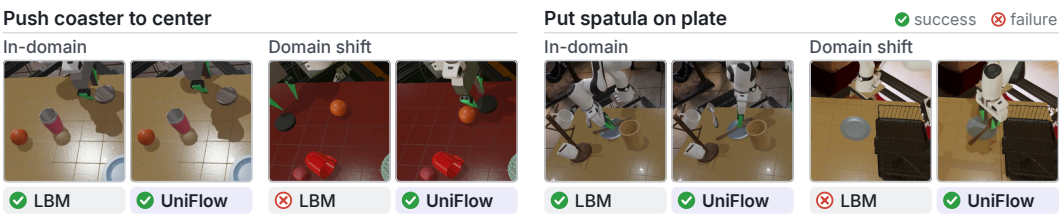
4.1 Benchmarks

- **LBM benchmark.** 13 simulated manipulation tasks (7 bimanual + 6 single-arm) from Barreiros et al. [3], set in three kitchen scenes and spanning pick-and-place, stacking, pushing, and reorientation. Each policy is evaluated *in-domain* and under *domain shift*, where lighting, background, textures, and camera parameters are perturbed; all policies share the same 100 initial conditions per task and condition.
- **RoboTwin.** Five bimanual tasks from RoboTwin 2.0 [12] (*handover block*, *lift pot*, *move can pot*, *pick dual bottles*, *turn switch*), with 100 demonstrations per task generated under randomized object poses, scene appearance, and lighting.
- **Real world.** Four bimanual tasks on the YAM platform, observed by a head and two wrist stereo cameras (Figure 4): *Plate Handoff* (handing a plate from one arm to the other and placing it into a rack), *Cup Stacking* (stacking two cups one inside the other in the rack), *Spoon Sort* (placing spoons one at a time into the rack's spoon holder), and *Plate Placement* (laying a plate flat in the rack, precisely on top of any existing stack), with about 50 teleoperated demonstrations per task.

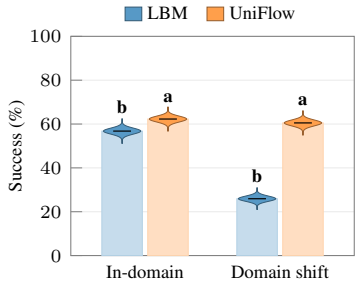
Unless otherwise specified, one multi-task checkpoint is trained per model per benchmark and evaluated per task. In the real world, each policy is evaluated with 20 rollouts per task (25 for Plate Placement), run in alternating sequential sets without matching initial conditions across methods; a human evaluator scores each rollout with a partial-credit rubric, and we report this normalized task score and the strict full-success rate. Task, data, and evaluation details are in App. D.

4.2 Baseline Models

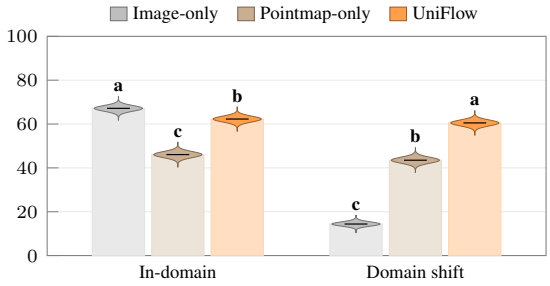
LBM (4-camera): A 4-camera variant of the LBM architecture [3], an RGB-based diffusion policy that consumes the same four camera views as UniFlow; our main baseline on the LBM benchmark. $\boldsymbol{\pi_{0.5}}$**:** The generalist VLA [11], fine-tuned on each benchmark's training set; our main RGB-only generalist baseline on RoboTwin and YAM. **ManiFlow:** A flow-matching policy [2] that takes colored point clouds as input and encodes them with a from-scratch 3D encoder; our main 3D baseline on RoboTwin. **UniFlow (variants).** The full joint 2D+3D model described in Section 3. On LBM, we additionally train two single-modality variants to isolate the contribution of each visual stream: **UniFlow Image-only** and **UniFlow Pointmap-only**, which remove the pointmap and RGB streams, respectively. Compute is matched across the three UniFlow variants. Compute budgets for all methods and UniFlow's optimizer and training-schedule settings are reported in the Appendix.



(a) Example rollouts in-domain and under domain shift



(b) LBM vs. UniFlow



(c) UniFlow modality ablation

Figure 5: **LBM benchmark under domain shift.** (a) Example rollouts in-domain and under domain shift, with the outcome for LBM and UniFlow. (b, c) Success over the 13 tasks: bars show the trial-weighted mean, violins the Beta posterior, and policies that share no letter differ at the 95% confidence level (App. D.5). From in-domain to domain shift, the success rate drops by 1.8 points for UniFlow, 2.6 for Pointmap-only, 30.9 for LBM, and 52.8 for Image-only.

5 Results

5.1 LBM Simulation Results

Table 1 compares UniFlow, UniFlow Image-only, UniFlow Pointmap-only, and LBM across the 13 LBM benchmark tasks in both *in-domain* and *domain-shift* settings, and Figure 5 summarizes the averages. Under domain shift, which demands robustness to lighting, background, and camera-position changes, UniFlow outperforms all other methods on 11 of 13 tasks and is competitive with UniFlow Pointmap-only on the remaining two—evidence that fusing 2D and 3D information generalizes better to these shifts than either image-based (UniFlow Image-only, LBM) or pointmap-based (UniFlow Pointmap-only) models alone. In-domain results are more mixed: the best method varies by task, and UniFlow Pointmap-only is never the best. When taking the sample-weighted mean over the per-task success rates, UniFlow scores higher than UniFlow Pointmap-only, demonstrating the value of adding a semantic stream to a geometric one.

The single-modality ablations have complementary strengths: UniFlow Image-only is generally stronger in-domain and UniFlow Pointmap-only under domain shift, motivating their fusion. On average, UniFlow trails the image-only variant in-domain, but the image-only variant also suffers the most drastic drop under domain shift of all four methods.

Domain shift by axis. Since pointmaps are camera-centered, we also apply each domain-shift axis in isolation (Table 10, App. E.5). Relative to the in-domain average, the appearance axes (lighting, texture, background) change UniFlow's success rate by $+0.1$ to -2.4 points and the camera axis by -0.8 points, so UniFlow is robust to both appearance and camera perturbations in this benchmark.

5.2 RoboTwin Simulation Results

Table 2a reports per-task success rates on the five RoboTwin tasks. UniFlow outperforms both $\pi_{0.5}$ and ManiFlow on three of five tasks (*Handover block*, *Lift pot*, *Move can pot*), is second on *Pick dual bottles*, and is third on *Turn switch*. These results suggest that UniFlow is strongest on tasks

Table 1: LBM benchmark: per-task success rate (%) of the LBM baseline and the UniFlow modality ablation. Cells target 100 rollouts; †: 95–99 completed due to software crashes. **Bold**: best per task and condition; **Average**: trial-weighted.

	In-domain				Domain shift			
Task	LBM	Img-only	Pmap-only	UniFlow	LBM	Img-only	Pmap-only	UniFlow
PlaceAppleFromBowlIntoBin	72.0	75.8†	67.0	**77.0**	52.0	32.3†	61.0	**73.0**
PlaceFruitFromBowlIntoBin	77.8†	**82.8**†	80.8†	82.0	54.5†	19.2†	**76.0**	72.0
PutRedBellPepperInBin	79.0	**85.0**	74.0	81.0	54.0	47.0	72.0	**84.0**
PutSpatulaOnPlateFromDryingRack	76.0	**77.0**	57.6†	74.0	22.4†	12.6†	66.0	**82.8**†
PutSpatulaOnPlateFromTable	62.0	75.0	44.0	**75.8**†	47.0	51.0	39.0	**76.0**
StackPlatesOnTableFromDryingRack	**93.0**	79.0	77.0	79.0	46.0	7.1†	**74.0**	70.0
StoreCerealBoxUnderShelf	**71.0**	62.6†	40.4†	**71.0**	17.0	15.5†	45.0	**70.0**
PlaceCupByCoaster	10.0	**52.0**	10.0	33.0	0.0	0.0	18.0	**36.0**
PushCoasterToCenterOfTable	52.0	55.0	32.3†	**60.0**	1.0	0.0†	25.0	**62.0**
PutBananaOnSaucer	2.0	**14.0**	7.0	11.0	0.0	0.0	3.0	**19.0**
PutMugOnSaucer	30.3†	**74.0**	23.0	44.0	3.0†	1.0	14.0	**39.0**
PutSpatulaInUtensilCrock	**80.0**	55.0	38.0	57.0	39.0	1.0†	35.4†	**48.0**
TurnCupUpsideDown	33.0	**86.0**	48.0	65.0	1.0	0.0	37.0	**55.0**
Average	56.8	**67.2**	46.1	62.3	25.9	14.4	43.5	**60.5**

Table 2: Results on RoboTwin and in the real world (%). (a) Success rate of $\pi_{0.5}$ [11], ManiFlow [2], and UniFlow on five RoboTwin [12] bimanual tasks, 100 rollouts per task. (b) Normalized task score (Score, with partial credit) and strict full-success rate (SR) on four YAM tasks, 20 rollouts per task (25 for Plate Placement); only the Plate Handoff difference in SR is statistically separable at these trial counts. **Bold**: best per task and metric; **Average**: weighted by rollout count.

(a) RoboTwin

Task	$\pi_{0.5}$	ManiFlow	UniFlow
Handover block	28.0	43.0	**54.0**
Lift pot	56.0	22.0	**67.0**
Move can pot	24.0	19.0	**27.0**
Pick dual bottles	34.0	**54.0**	47.0
Turn switch	**40.0**	28.0	27.0
Average	36.4	33.2	**44.4**

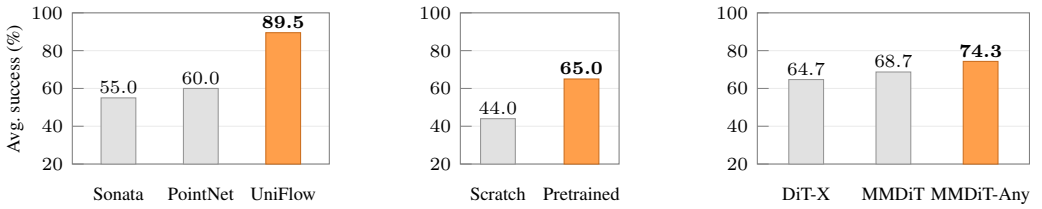
(b) Real world (YAM)

	Score		SR	
Task	$\pi_{0.5}$	UniFlow	$\pi_{0.5}$	UniFlow
Plate Handoff	22.50	**75.00**	0.0	**60.0**
Cup Stacking	**66.25**	51.25	**30.0**	5.0
Spoon Sort	16.00	**28.00**	15.0	**20.0**
Plate Placement	**80.00**	78.00	**60.0**	**60.0**
Average	48.18	**59.24**	28.2	**37.6**

requiring precise 3D placement, where the pointmap stream adds information beyond $\pi_{0.5}$'s RGB inputs and pretrained 2D backbones on pointmaps outperform ManiFlow's from-scratch 3D encoder.

5.3 Real-World: YAM Bimanual Tasks

Figure 16 in App. F shows representative rollouts of UniFlow on the four YAM real-world tasks, illustrating the range of bimanual behaviors, including mid-air handoff, stacking, and placement. In Table 2b, UniFlow scores higher than $\pi_{0.5}$ on *Plate Handoff* and *Spoon Sort*, is competitive on *Plate Placement* (equal full-success rates), and underperforms on *Cup Stacking*; averaged over the tasks, it achieves a higher normalized score and full-success rate. At the current trial counts, however, only the Plate Handoff difference in full-success rate (60.0% vs. 0.0%) is statistically separable. $\pi_{0.5}$ scores higher than UniFlow on the two real-world tasks involving placing objects onto one another, suggesting that visual cues alone may suffice for these placement tasks. In contrast, UniFlow is more successful on the tasks in our real-world benchmark requiring a bimanual handoff or precise grasp and placement, pointing to the benefit of our multi-modal architecture. **Failure analysis.** On Cup Stacking, where UniFlow has its lowest full-success rate (5.0%), it usually grasps the cup and localizes the correct slot but leaves the cup only partially aligned or bumps it, causing it to fall. $\pi_{0.5}$ never completes Plate Handoff: its right gripper repeatedly misses the thin plate edge during the handover and drops the plate (9 of 20 handovers succeed), and it never places the plate in the rack.



(a) 3D encoder (b) Pointmap-encoder pretraining (c) Multi-modal fusion architecture

Figure 6: **Ablations.** (a) Different 3D encoders. (b) Pointmap encoder with and without pretraining. (c) Different multi-modal fusion architectures. Experimental details are in App. E.

5.4 Ablations: 3D Representation and Multi-Modal Fusion

3D representation. Replacing only the pointmap stream with a point-cloud encoder lowers success, whether that encoder is trained from scratch or pretrained (Figure 6a); with 256 points, which may not be optimal for each encoder, the Sonata-pretrained point transformer is comparable to the ManiFlow baseline in the same setting (54.0%) and below a from-scratch PointNet. In a separate experiment, we additionally replaced the pointmap stream with PointNet++ [15] (8.0%), RandLA-Net [27] (42.0%), and a pretrained Uni3D [19] (22.0%), versus 83.0% for UniFlow (App. E.2). Pretrained point-cloud encoders thus did not close the gap in our experiments. These results suggest that a pretrained pointmap encoder is well-suited for policy prediction; we hypothesize that this is due to the favorable alignment between the image and pointmap spaces.

Pretraining. In a separate experiment with architecture, data, and compute fixed, using a pretrained ConvNeXt V2 pointmap encoder, rather than training it from scratch, raises UniFlow's success rate on LBM *PutRedBellPepperInBin* from 44.0% to 65.0% (Figure 6b): pointmaps are compelling in part *because* they let the 3D stream reuse pretrained image encoders.

Multi-modal fusion. We compare MMDiT-Any with *DiT-X*, where an action-only stream cross-attends to the observation tokens as in ManiFlow, and with *MMDiT*, which follows the original two-stream design: RGB and pointmap tokens are concatenated into one observation stream with shared modulation, attention projections, and feed-forward parameters. MMDiT-Any is best on average (Figure 6c) and on each task; its gain over MMDiT isolates the effect of giving each modality its own modulation, attention projections, and feed-forward parameters.

Capacity and training budget. A smaller UniFlow, whose fusion transformer has fewer parameters than ManiFlow's DiT-X but more parameters in total, reaches 77.0%, and the full-size model trained for half as many epochs reaches 81.0%, both above ManiFlow at 54.0% (App. E.3).

6 Conclusions and Limitations

We have presented UniFlow, which represents 3D observations as pointmaps so that pretrained 2D backbones encode both RGB and geometry, and fuses the two streams with per-modality modulation. Across 13 LBM tasks, UniFlow loses only 1.8 points under domain shift, whereas the RGB-only LBM policy loses 30.9, and it outperforms $\pi_{0.5}$ on average on RoboTwin and on four real-world bimanual tasks. These results suggest that treating 3D as an image is a promising way to bring large-scale 2D pretraining to robust multi-modal robot policies.

Limitations. UniFlow builds pointmaps from registered depth, and we have not yet evaluated its sensitivity to depth noise, missing depth, calibration errors, or transparent and reflective surfaces. On LBM, its advantage is largest under domain shift; in-domain, the image-only variant is slightly stronger on average. Modality collapse can still occur under certain hyperparameter settings; larger pretrained encoders such as DINOv3 [28] may help mitigate it. Future work also includes scaling up UniFlow pretraining on larger datasets and predicting pointmaps with a foundation model.

Acknowledgments

This work was supported by the Toyota Research Institute.

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*, 44(10-11):1684–1704, 2025.

[2] G. Yan, J. Zhu, Y. Deng, S. Yang, R.-Z. Qiu, X. Cheng, M. Memmel, R. Krishna, A. Goyal, X. Wang, et al. ManiFlow: A general robot manipulation policy via consistency flow training. *arXiv preprint arXiv:2509.01819*, 2025.

[3] J. Barreiros, A. Beaulieu, A. Bhat, R. Cory, E. Cousineau, H. Dai, C.-H. Fang, K. Hashimoto, M. Z. Irshad, M. Itkina, et al. A careful examination of large behavior models for multitask dexterous manipulation. *Science Robotics*, 11(113):eaea6201, 2026.

[4] J. Wang, M. Leonard, K. Daniilidis, D. Jayaraman, and E. Hu. Evaluating pi0 in the wild: Strengths, problems, and the future of generalist robot policies. https://penn-pal-lab.github.io/Pi0-Experiment-in-the-Wild/, 2025.

[5] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. *arXiv preprint arXiv:2403.03954*, 2024.

[6] M. Shridhar, L. Manuelli, and D. Fox. Perceiver-Actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023.

[7] W. Shen, G. Yang, A. Yu, J. Wong, L. P. Kaelbling, and P. Isola. Distilled feature fields enable few-shot language-guided manipulation. *arXiv preprint arXiv:2308.07931*, 2023.

[8] X. Jia, Q. Wang, A. Wang, H. Wang, B. Gyenes, E. Gospodinov, X. Jiang, G. Li, H. Zhou, W. Liao, et al. Pointmappolicy: Structured point cloud processing for multi-modal imitation learning. *Advances in Neural Information Processing Systems*, 38:160188–160216, 2026.

[9] P. Esser, S. Kulal, A. Blattmann, R. Entezari, J. Müller, H. Saini, Y. Levi, D. Lorenz, A. Sauer, F. Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

[10] W. Peebles and S. Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

[11] Physical Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

[12] T. Chen, Z. Chen, B. Chen, Z. Cai, Y. Liu, Z. Li, Q. Liang, X. Lin, Y. Ge, Z. Gu, et al. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

[13] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[14] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[15] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[16] Y. Ze, G. Yan, Y.-H. Wu, A. Macaluso, Y. Ge, J. Ye, N. Hansen, L. E. Li, and X. Wang. GNFactor: Multi-task real robot learning with generalizable neural feature fields. In *Conference on robot learning*, pages 284–301. PMLR, 2023.

[17] G. Yan, Y.-H. Wu, and X. Wang. Dnact: Diffusion guided multi-task 3d policy learning. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9464–9471. IEEE, 2025.

[18] Y. Li, G. Yan, A. Macaluso, M. Ji, X. Zou, and X. Wang. Integrating lmm planners and 3d skill policies for generalizable manipulation. *arXiv preprint arXiv:2501.18733*, 2025.

[19] J. Zhou, J. Wang, B. Ma, Y.-S. Liu, T. Huang, and X. Wang. Uni3D: Exploring unified 3D representation at scale. In *International Conference on Learning Representations*, volume 2024, pages 46766–46782, 2024.

[20] X. Wu, D. DeTone, D. Frost, T. Shen, C. Xie, N. Yang, J. Engel, R. Newcombe, H. Zhao, and J. Straub. Sonata: Self-supervised learning of reliable point representations. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22193–22204. IEEE, 2025.

[21] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud. DUSt3R: Geometric 3D vision made easy. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20697–20709, 2024.

[22] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.

[23] W. Wang, D. Tran, and M. Feiszli. What makes training multi-modal classification networks hard? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12695–12705, 2020.

[24] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[25] S. Woo, S. Debnath, R. Hu, X. Chen, Z. Liu, I. S. Kweon, and S. Xie. ConvNeXt V2: Co-designing and scaling ConvNets with masked autoencoders. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16133–16142, 2023.

[26] K. Frans, D. Hafner, S. Levine, and P. Abbeel. One step diffusion via shortcut models. In *International Conference on Learning Representations*, volume 2025, pages 34668–34684, 2025.

[27] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham. RandLA-Net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11108–11117, 2020.

[28] O. Siméoni, H. V. Vo, M. Seitzer, F. Baldassarre, M. Oquab, C. Jose, V. Khalidov, M. Szafraniec, S. Yi, M. Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

[29] R. Tedrake and the Drake Development Team. Drake: Model-based design and verification for robotics, 2019. URL `https://drake.mit.edu`.

Appendix

A Overview

This appendix provides the full experimental details behind the results in the main paper:

- **App. B — Detailed results breakdown:** per-task success rates on the LBM benchmark for all policies, with the Bayesian posteriors and statistical-significance tests behind the aggregate numbers.
- **App. C — Implementation details:** visual encoders, data augmentation, the MMDiT-Any backbone, language conditioning, the training recipe, modality masking, and background on flow matching, AdaLN, MMDiT, and the pretrained backbones.
- **App. D — Dataset and evaluation:** tasks and demonstration budgets in simulation and the real world, the real-world scoring rubric, camera configurations, the domain-shift protocol, and the statistical methodology.
- **App. E — Additional results:** ablations on per-modality modulation, the choice of 3D encoder (including a pretrained point-cloud encoder), the initialization of the pointmap encoder, model capacity, and domain shift by axis.
- **App. F and App. G — Qualitative examples:** rollout comparisons in the real world and on the LBM benchmark, respectively.
- **App. H — Compute:** training hardware, durations, and reproducibility information.

B Additional Figures and Detailed Results Breakdown

Table 3: Per-task success rate (%) for LBM vs. UniFlow, evaluated in-domain and under domain shift. Mean over up to 100 rollouts per task; a superscript (n) gives the number of completed rollouts where it is below 100 (cells marked † in Table 1). **Bold** marks the higher value per (task, condition).

	In-domain		Domain shift	
Task	LBM	UniFlow	LBM	UniFlow
PlaceAppleFromBowlIntoBin	72.0	**77.0**	52.0	**73.0**
PlaceFruitFromBowlIntoBin	77.8$^{(99)}$	**82.0**	54.5$^{(99)}$	**72.0**
PutRedBellPepperInBin	79.0	**81.0**	54.0	**84.0**
PutSpatulaOnPlateFromDryingRack	**76.0**	74.0	22.4$^{(98)}$	**82.8**$^{(99)}$
PutSpatulaOnPlateFromTable	62.0	**75.8**$^{(99)}$	47.0	**76.0**
StackPlatesOnTableFromDryingRack	**93.0**	79.0	46.0	**70.0**
StoreCerealBoxUnderShelf	**71.0**	**71.0**	17.0	**70.0**
PlaceCupByCoaster	10.0	**33.0**	0.0	**36.0**
PushCoasterToCenterOfTable	52.0	**60.0**	1.0	**62.0**
PutBananaOnSaucer	2.0	**11.0**	0.0	**19.0**
PutMugOnSaucer	30.3$^{(99)}$	**44.0**	3.0$^{(99)}$	**39.0**
PutSpatulaInUtensilCrock	**80.0**	57.0	39.0	**48.0**
TurnCupUpsideDown	33.0	**65.0**	1.0	**55.0**
Average	56.8	**62.3**	25.9	**60.5**

Per-task statistical comparison. We complement the mean success rates of Tables 3 and 4 with per-task Bayesian posteriors and a formal multiple-comparison test. Each (policy, task, condition) cell is evaluated against a planned budget of 100 LBM benchmark scenarios with deterministic seeds, so that all four policies are tested under matched initial conditions. Per-episode binary outcomes are read from each rollout's `demonstration_N/summary.yaml`; rollouts whose summary is absent (catastrophic crash) are dropped, matching the LBM SR denominator and leaving 95–100 completed episodes per cell across the 104 cells. While the root cause of the dropped trials is unknown, it is possible that certain policy rollouts may be more prone to causing simulation issues.

Table 4: Per-task success rate (%) for the UniFlow modality ablation. Img-only and Pmap-only are single-stream variants; UniFlow is the full masked + modulated joint model (mask 0.1 / 0.1). Superscripts (n) as in Table 3. **Bold** marks the best per (task, condition).

	In-domain			Domain shift		
Task	Img-only	Pmap-only	UniFlow	Img-only	Pmap-only	UniFlow
PlaceAppleFromBowlIntoBin	75.8$^{(99)}$	67.0	**77.0**	32.3$^{(99)}$	61.0	**73.0**
PlaceFruitFromBowlIntoBin	**82.8**$^{(99)}$	80.8$^{(99)}$	82.0	19.2$^{(99)}$	**76.0**	72.0
PutRedBellPepperInBin	**85.0**	74.0	81.0	47.0	72.0	**84.0**
PutSpatulaOnPlateFromDryingRack	**77.0**	57.6$^{(99)}$	74.0	12.6$^{(95)}$	66.0	**82.8**$^{(99)}$
PutSpatulaOnPlateFromTable	75.0	44.0	**75.8**$^{(99)}$	51.0	39.0	**76.0**
StackPlatesOnTableFromDryingRack	**79.0**	77.0	**79.0**	7.1$^{(99)}$	**74.0**	70.0
StoreCerealBoxUnderShelf	62.6$^{(99)}$	40.4$^{(99)}$	**71.0**	15.5$^{(97)}$	45.0	**70.0**
PlaceCupByCoaster	**52.0**	10.0	33.0	0.0	18.0	**36.0**
PushCoasterToCenterOfTable	55.0	32.3$^{(99)}$	**60.0**	0.0$^{(99)}$	25.0	**62.0**
PutBananaOnSaucer	**14.0**	7.0	11.0	0.0	3.0	**19.0**
PutMugOnSaucer	**74.0**	23.0	44.0	1.0	14.0	**39.0**
PutSpatulaInUtensilCrock	55.0	38.0	**57.0**	1.0$^{(98)}$	35.4$^{(99)}$	**48.0**
TurnCupUpsideDown	**86.0**	48.0	65.0	0.0	37.0	**55.0**
Average	**67.2**	46.1	62.3	14.4	43.5	**60.5**

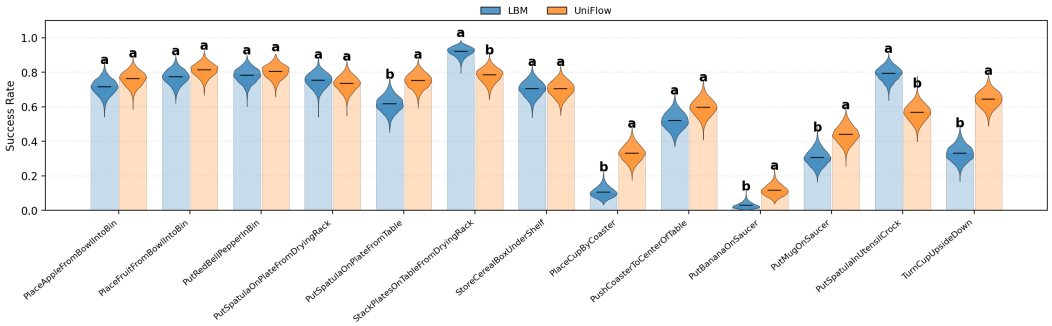


Figure 7: LBM vs. UniFlow: per-task in-domain success-rate posteriors and compact letter displays.

The per-task violins in Figures 7–12 visualize the posterior $\mathrm{Beta}(1{+}k,\ 1{+}n{-}k)$ over the true per-task success rate, where k and n are the observed successes and completed episodes. Compact letter displays (CLDs) are computed with the sequentialized Barnard STEP test on the real ordered binary sequences (global confidence 0.95, `max_sample_size_per_model` $=100$, and `shuffle=False` because within a single task the outcomes are i.i.d. across independent scenario seeds); two policies share a letter if and only if the test fails to reject equality at the Bonferroni-corrected per-pair level.

UniFlow vs. LBM. UniFlow dominates the LBM 4-camera baseline in both regimes (Figures 7 and 8, with the per-task numbers in Table 3 and the aggregated view in Figure 13). In-domain it wins 9 of 13 tasks (with one tie); under domain shift it wins all 13. The three tasks where LBM is ahead in-domain — StackPlatesOnTableFromDryingRack (93 vs. 79), PutSpatulaInUtensilCrock (80 vs. 57), and PutSpatulaOnPlateFromDryingRack (76 vs. 74) — are precision placements where strong RGB features alone already saturate. The average gap is modest in-domain (56.8 vs. 62.3) because LBM and UniFlow both clear $\sim 70\%$ on the easier bimanual pick-and-place tasks, but widens sharply under domain shift (25.9 vs. 60.5): LBM nearly collapses on the five tasks where randomization most stresses visual perception — PlaceCupByCoaster, PushCoasterToCenterOfTable, PutBananaOnSaucer, PutMugOnSaucer, and TurnCupUpsideDown (all $\leq 3\%$) — while UniFlow retains at least 19% success on every one of them.

Modality ablation. The single-modality ablations (Figs. 9 and 10, Table 4, and the aggregated view in Fig. 14) cleanly separate the two complementary failure modes that motivate the fused

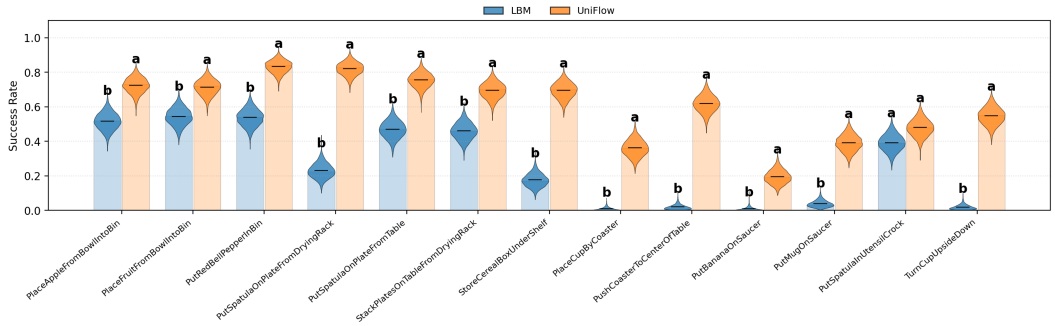


Figure 8: LBM vs. UniFlow: per-task domain-shift success-rate posteriors and compact letter displays.

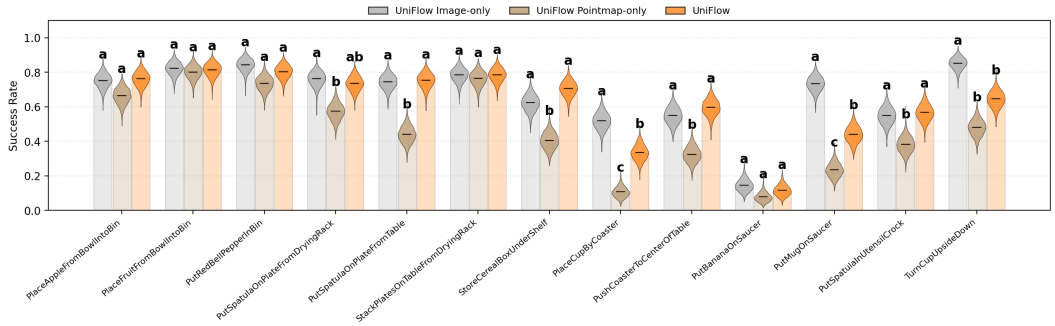


Figure 9: UniFlow modality ablation: in-domain per-task posteriors and compact letter displays for the Image-only baseline, the Pointmap-only baseline, and the full UniFlow image+pointmap fusion.

design. In-domain, the image-only baseline attains the highest average (67.2), slightly above full UniFlow (62.3) and well above the pointmap-only baseline (46.1): when the train- and test-time visual statistics agree, RGB features carry most of the semantics needed for manipulation. Under domain shift this advantage inverts: image-only drops by 52.8 points to 14.4, while pointmap-only, which encodes the scene geometrically and is largely invariant to lighting, tabletop texture, and scene-intrinsic perturbations, loses only 2.6 points and ends ahead at 43.5. The fused UniFlow inherits nearly all of the pointmap modality's shift robustness (-1.8 points, ending at 60.5) while sacrificing only ≈ 5 points of in-domain headroom relative to image-only, in exchange for $+46$ points under shift. No single-modality baseline is competitive in both regimes; only the joint image + pointmap UniFlow model is.

Four-way comparison. Figures 11 and 12 place all four policies on common per-task panels. Adding LBM enlarges the multiple-comparison family from 3 to 6 pairs, tightening the per-pair Bonferroni threshold from $\alpha = 0.05/3$ to $\alpha = 0.05/6$ and consequently softening some per-task CLD calls relative to the two- and three-way figures above. The average ordering nonetheless preserves the picture from Tables 3 and 4: in-domain, the image-only baseline (67.2) and full UniFlow (62.3) lead, with LBM (56.8) and the pointmap-only baseline (46.1) trailing; under domain shift, UniFlow (60.5) stands alone at the top, pointmap-only (43.5) is second, and LBM (25.9) and image-only (14.4) collapse together at the bottom. Per task, UniFlow is in the top statistical group on all 13 tasks under shift and on 8 of 13 in-domain (the image-only baseline: 12 in-domain, none under shift), and it is the only policy that ranks in the top two on average under both conditions.

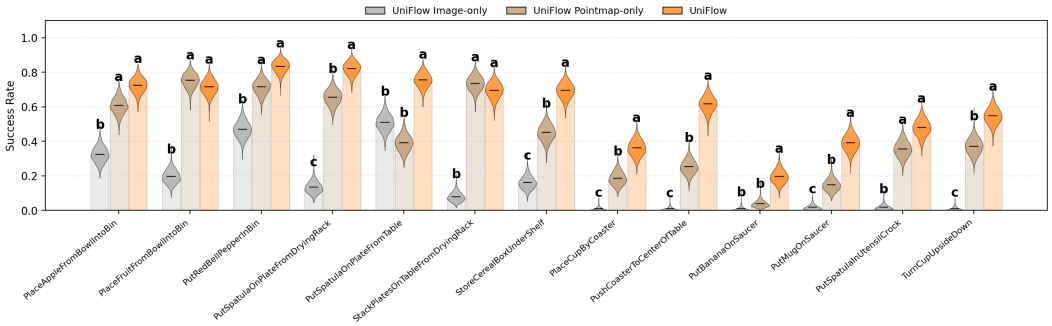


Figure 10: UniFlow modality ablation: domain-shift per-task posteriors and compact letter displays across the same three variants.

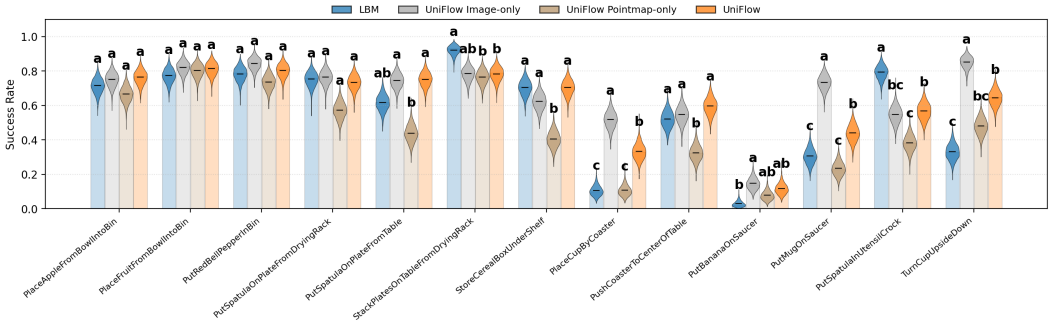


Figure 11: Four-way in-domain comparison across LBM, the two UniFlow single-modality ablations, and full UniFlow.

C Implementation Details

C.1 Visual Encoders

RGB encoder. A pretrained CLIP ViT-B/16 backbone (`vit_base_patch16_clip_224.openai` from `timm`). Images are resized to 224×224 and encoded into a spatial feature map that is flattened into a token sequence.

Pointmap encoder. A pretrained ConvNeXt V2 Tiny backbone. The pointmap is treated as a 3-channel image, clipped to a per-camera bounding box, and normalized to $[-1, 1]$ per channel before being passed to the pointmap backbone.

DINOv3 as the 2D encoder. In our real-world experiments, we additionally found that using DINOv3 [28] as the pretrained 2D encoder brings better performance and can be more stable.

C.2 Data Augmentation

We follow the $\pi_{0.5}$ recipe [11] for RGB augmentation:

- **Geometric augmentation** (*non-wrist cameras only*): random crop to 95% of the image size, resize back to the original size, random in-plane rotation in $[-5°, +5°]$.
- **Color augmentation** (all cameras): color jitter with brightness 0.3, contrast 0.4, saturation 0.5, hue 0.0.

Wrist cameras skip geometric augmentation because their optical centers are rigidly attached to the grippers and the policy must learn the camera-to-end-effector geometry. Pointmaps are not augmented because their values are physical coordinates.

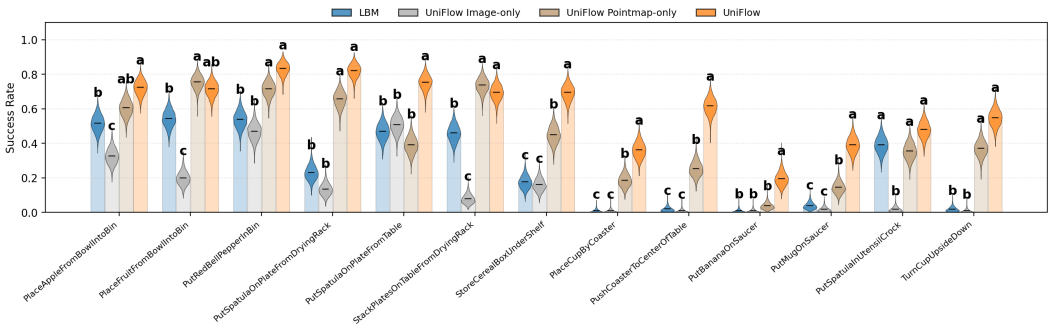


Figure 12: Four-way domain-shift comparison across the same four policies.

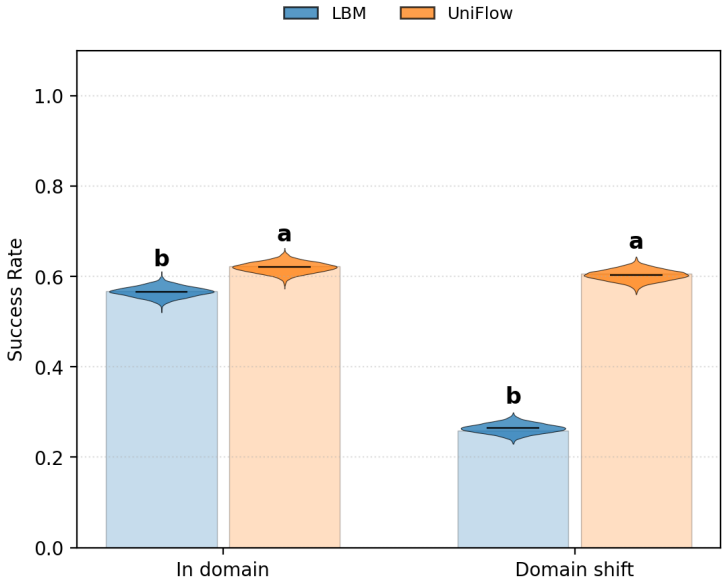


Figure 13: LBM vs. UniFlow aggregated across 13 tasks, evaluated in-domain and under domain shift. Bars show mean success rate; violins show the Beta posterior. Compact letter displays (**a** / **b**) indicate statistical significance — groups that share no letter are distinguishable at the 95% confidence level.

C.3 MMDiT-Any Backbone

The backbone consists of $N = 8$ MMDiT-Any joint-attention blocks, hidden dimension $D = 768$, and 12 attention heads. Each block applies a per-modality feed-forward network with an MLP expansion ratio of 4 and uses GELU activations. Attention is computed jointly over the concatenated modality tokens using a flash-attention (scaled-dot-product) kernel.

Each block applies (i) a joint self-attention over the concatenation of all modality streams, with per-modality query, key, and value projections; (ii) per-modality feed-forward networks; and (iii) per-modality adaptive normalization (AdaLN) with separate scale/shift parameters per modality and per camera. Global conditioning includes the diffusion timestep embedding, the consistency target step size, proprioception, and a language embedding, which are mapped through modality-specific MLPs to produce these AdaLN parameters. Figure 3 shows one block.

Figure 14: UniFlow modality ablation aggregated across 13 tasks. Image-only and Pointmap-only variants vs. full UniFlow. The single-modality variants are individually strong on exactly one of the two conditions; only the joint masked + modulated model is competitive on both.

C.4 Language Encoder

When language conditioning is enabled, raw text prompts are encoded with the CLIP text encoder. By default the pooled embedding is used and injected as global conditioning; per-token embeddings are also supported and would be appended to the token sequence as context.

C.5 Training Recipe

Objective. Consistency flow matching [2] with flow ratio 0.75 and consistency ratio 0.25 within each mini-batch. The flow term uses $\Delta t = 0$; the consistency term uses $\Delta t \sim \mathcal{U}[0, 1]$. We sample t continuously — beta distribution for the flow term and uniform for the consistency term in our default configuration — following ManiFlow.

Optimizer and schedule. All UniFlow models are trained with AdamW (β_1=0.9, β_2=0.95, ϵ=10^{-8}), peak learning rate 10^{-4}, and weight decay 10^{-3}, under a cosine learning-rate schedule with linear warmup (2,000 steps on the LBM benchmark; 500 steps on RoboTwin and the real-world tasks). We maintain an exponential moving average (EMA) of the weights with decay $1-(1+s)^{-0.75}$ at step s, capped at 0.9999, and use the EMA weights for all evaluations. Training durations and hardware are listed in App. H.

C.6 Modality Masking

For each modality $m \in \{\text{rgb}, \text{pointmap}\}$ we sample a Bernoulli mask at rate $p_m = 0.1$ at training time, and each camera of each modality is masked independently.

C.7 Inference

At deployment, an action chunk is generated by sampling $x_0 \sim \mathcal{N}(0, \mathbf{I})$ and integrating the learned velocity field from $t = 0$ to $t = 1$ with Euler steps. We use receding-horizon execution: at each control cycle the policy predicts a chunk of H actions, executes n_{exec} of them, and re-plans.

C.8 Preliminaries

Flow matching and consistency flow matching. Flow matching [13] learns a velocity field $v_\theta(x_t, t)$ that transports noise $x_0 \sim \mathcal{N}(0, \mathbf{I})$ to data x_1 along the linear path $x_t = (1-t)\,x_0 + t\,x_1$, $t \in [0, 1]$, by regressing the target velocity $x_1 - x_0$; samples are generated by integrating v_θ from $t = 0$ to $t = 1$ with an ordinary differential equation (ODE) solver such as the Euler method. Consistency flow matching, as used in ManiFlow [2], additionally conditions the network on a step size Δt and trains it to predict the average velocity over a finite interval [26], so that the velocity field can be integrated with large steps (Section 3.3 and App. C.7).

Adaptive layer normalization (AdaLN) and MMDiT. Diffusion transformers [10] inject global conditioning, such as the diffusion timestep, through AdaLN: a small network maps a conditioning vector c to per-channel scale and shift parameters that modulate the layer-normalized activations of all tokens in a block, and to gates that scale each residual branch. The multi-modal diffusion transformer (MMDiT) [9] processes two token streams (text and image) with separate weights, including separate AdaLN modulation, and lets them interact through a joint self-attention over the concatenated sequence. UniFlow's MMDiT-Any generalizes this design to an arbitrary number of streams (Section 3.2).

Pretrained backbones. CLIP ViT-B/16 [24] is a Vision Transformer operating on 16×16 image patches, pretrained with contrastive image–text learning; ConvNeXt V2 [25] is a convolutional network co-designed with masked-autoencoder pretraining.

D Dataset and Evaluation

D.1 Tasks and Demonstrations

D.1.1 Simulation Demonstrations

LBM benchmark (13 tasks). We use 13 tasks from the LBM benchmark [3], built on the Drake simulator [29] and set in three simulated kitchen scenes: a *shelf* scene (fruit bowl, bin, shelf, cereal box), a *drying-rack* scene (plates, spatulas, drying rack, utensil crock), and a *breakfast-table* scene (cup, mug, saucer, coaster). Seven tasks are bimanual (PLACEAPPLEFROMBOWLINTOBIN, PLACEFRUITFROMBOWLINTOBIN, PUTREDBELLPEPPERINBIN,PUTSPATULAONPLATEFROMDRYINGRACK, PUTSPATULAONPLATEFROMTABLE, STACKPLATESONTABLEFROMDRYINGRACK, and STORECEREALBOXUNDERSHELF) and six are single-arm (PLACECUPBYCOASTER, PUSHCOASTERTOCENTEROFTABLE, PUTBANANAONSAUCER, PUTMUGONSAUCER, PUTSPATULAINUTENSILCROCK, and TURNCUPUPSIDEDOWN). Tasks span pick-and-place, stacking, pushing, and reorientation, and each policy is conditioned on the task's natural-language instruction provided by the benchmark. All policies are trained on the simulation demonstrations released with the benchmark.

RoboTwin (5 tasks). We use five bimanual tasks from RoboTwin [12]: *handover block*, *lift pot*, *move can pot*, *pick dual bottles*, and *turn switch*, with 100 demonstrations per task (500 in total). Demonstrations are generated by the benchmark's automated expert-data pipeline under its domain-randomized configuration (randomized object poses, scene appearance, and lighting). One multi-task checkpoint per method is trained on the combined set and evaluated per task.

D.1.2 Real-World Demonstrations

We evaluate on four tasks on the bimanual YAM platform: **Plate Handoff** (transfer a plate from one arm to the other and place it into the rack), **Cup Stacking** (pick up two cups and stack them one inside the other in the rack), **Plate Placement** (pick up a plate and lay it flat into the rack, precisely on top of any existing stack), and **Spoon Sort** (pick up spoons one at a time and place each into the spoon-holder section of the rack).

All tasks are controlled at 30 Hz, with episodes spanning roughly 540 control steps on average (about 18 s per episode). Between episodes we randomize the starting positions of all task-relevant objects, placing them at varied locations within the arms' reachable workspace; the rack is also repositioned across episodes but always kept within the right half of the table.

All real-world demonstrations are collected via human teleoperation. For each of the four tasks we record roughly 50 episodes and discard corrupted trajectories (approximately two per task), leaving 46–51 demonstrations per task and 191 in total, or roughly 57 minutes of interaction data. The four tasks are concatenated into a single dataset, with episode indices assigned in the order shown in Table 5; one unified model is trained on the combined set. This demonstration budget is held fixed across all variants.

Table 5: Per-task demonstration budget for the YAM real-world tasks. All four tasks are combined into a single 191-episode dataset used to train one unified model.

Task	Episode Range	# Episodes
Plate Handoff	0–50	51
Cup Stacking	51–97	47
Plate Placement	98–143	46
Spoon Sort	144–190	47
Total	0–190	**191**

D.2 Real-World Evaluation Criteria

Evaluation protocol. For the four real-world tasks, we evaluate each policy with $n{=}20$ rollouts per task, run in alternating sequential sets (e.g., 10 trials per method). Initial conditions were not matched across methods, though we made efforts to sample them fairly for each method. This is a limitation of our protocol: because object starting positions were not held fixed across trials or models, per-task numbers carry some variance from initial-condition differences rather than policy quality alone. For Plate Placement, the two methods' scores were close after 20 trials, so we collected 5 additional trials per method (25 per method).

A note on terminology. For the real-world tasks, the main paper reports two metrics. The *normalized task score* is obtained by scoring each rollout with a human evaluator under the partial-credit rubric below and dividing the credit earned by the maximum attainable credit. This score provides more granularity than a binary success rate — a rollout that completes one of two equally weighted sub-goals contributes 50% rather than 0 — and coincides with the success rate when no partial credit is awarded. The strict *full-success rate* (SR) counts a rollout as successful only if the entire task is completed. We use "normalized task score" throughout this appendix. All simulation results remain ordinary binary success rates.

- *Plate Handoff:* The task begins with a plate on the table. The robot must pick it up with one arm, transfer it to the other arm, and place it into the plate rack.

  +0.5 Handing the plate off between the two arms without dropping it.

  +0.5 Placing the plate into the rack.

  Maximum score: 1 point per rollout; $n{=}20$ rollouts $\Rightarrow$ 20 pt attainable in total.

- *Cup Stacking:* The task begins with two cups on the table. The robot's goal is to place both cups upright in the target configuration.

  +1 For each cup placed upright in the target position.

  +0.5 For each cup placed but left tipped over (in lieu of the upright credit).

  Maximum score: 2 points per rollout (1 per cup); $n{=}20$ rollouts $\Rightarrow$ 40 pt attainable in total.

- *Spoon Sort:* The task begins with N spoons on the table. The robot must sort each spoon into its target location. In our evaluation, 15 of the 20 rollouts use $N{=}1$ and 5 use $N{=}2$, for 25 attainable points in total.

  +1 For each spoon placed correctly in the target location.

  Maximum score: N points per rollout; 15 rollouts with $N{=}1$ and 5 with $N{=}2 \Rightarrow$ 25 pt attainable in total.

- *Plate Placement:* The task begins with 1 plate near the rack. The robot must place each plate into the rack.

  +0.5 For placing the plate in the rack, even if it is misaligned with the stack (Step 1).

  +0.5 Additionally, for stacking it flush on top of the existing stack (or onto an empty rack) (Step 2).

  Maximum score: 1 point per rollout; $n{=}25$ rollouts $\Rightarrow$ 25 pt attainable in total.

Each task is decomposed into ordered steps that mirror the sub-goals a successful rollout must complete, with the task's total credit split equally across its steps. Table 6 reports, for every step, the credit earned as a percentage of that step's *maximum* credit (Figure 15 plots the same values as fractions), averaged over the evaluation rollouts ($n{=}20$ per task; $n{=}25$ for Plate Placement) — e.g., 90.0 on a 0.5 pt step means 0.45 pt earned per rollout on average. Since steps are equally weighted, the normalized task score is simply the mean of its step scores, i.e., total credit earned over maximum attainable credit.

Plate Handoff (0.5 pt per step): Step 1 scores transferring the plate between arms without dropping it; Step 2 scores placing the handed-off plate into the rack. The gap between $\pi_{0.5}$ and UniFlow appears in both steps and is largest in Step 2, where the baseline never completes the placement. **Cup Stacking** (1 pt per step): Steps 1 and 2 score the first and second cup being placed upright in the target stacked position, with half credit for a cup that is placed but tipped over. **Spoon Sort** (single step, 1 pt per spoon): the score is the fraction of spoons sorted into the target holder, pooled over the 25 spoons presented across the 20 rollouts — 15 rollouts present a single spoon ($N{=}1$) and the remaining 5 present two ($N{=}2$), giving $15{\times}1 + 5{\times}2 = 25$ spoons and 25 pt of attainable credit in total ($4/25$ for $\pi_{0.5}$, $7/25$ for UniFlow). **Plate Placement** (0.5 pt per step): Step 1 scores laying the plate flat into the rack; Step 2 scores stacking it flush on the existing stack, so a placed but misaligned plate earns Step 1 only. See Figures 17–20 for representative rollouts.

D.3 Camera Configuration

D.3.1 Simulation Camera Configuration

On the LBM benchmark, both LBM and UniFlow use the same four cameras: two scene cameras (left and right) and the left- and right-wrist cameras. On RoboTwin, $\pi_{0.5}$ uses three cameras (head, left wrist, and right wrist), while ManiFlow and UniFlow use the head camera.

D.3.2 Real-World Camera Configuration

Our real-world setup uses three stereo cameras: a ZED 2i mounted in between two arms as head camera and two ZED M cameras mounted on the left and right wrists. Depth is computed via stereo matching on each device, and we take the left-eye image as the RGB input. All streams are recorded at 1280×720 resolution and 30 fps. Pointmaps are constructed in the camera frame of each respective camera.

D.4 Domain-Shift Protocol

The domain-shift condition follows the *Distribution Shift (Sim)* protocol of the LBM benchmark [3] and primarily perturbs scene appearance. Under the in-domain (nominal) condition, object poses and counts are drawn from predefined distributions with rejection sampling to avoid interpenetration, and a single directional light source has its color, intensity, and direction drawn from in-distribution

Table 6: Per-step breakdown of the real-world evaluation. Each (task, policy) cell is scored by a human evaluator over n=20 rollouts (n=25 for Plate Placement) under the rubric of App. D.2. **All values are percentages of the maximum credit attainable for that row**: step rows report the credit earned as a percentage of the step's maximum credit (in parentheses in the row label), averaged over rollouts; the task-level *normalized score* is total credit earned over maximum attainable credit, equal to the mean of the task's step rows since steps are equally weighted. The final row is the mean normalized score weighted by rollout count (20+20+20+25=85 rollouts per policy) — i.e., the rollout-count-weighted mean of per-task scores, not pooled raw credit across tasks, so each task is weighted by its number of rollouts (within Spoon Sort, credit is pooled over spoons). **Bold** marks the better value per row. Table 2b of the main paper reports these task-level values as normalized task scores (Score), alongside strict full-success rates (SR); see the terminology note in App. D.2.

Task	**Step (max credit; values in % of max)**	$\pi_{0.5}$	**UniFlow**
Plate Handoff	Step 1: Hand off plate between arms (0.5 pt)	45.0	**90.0**
	Step 2: Place plate into rack (0.5 pt)	0.0	**60.0**
	Normalized task score (of 20 pt)	22.5	**75.0**
Cup Stacking	Step 1: First cup upright (1 pt)	**67.5**	57.5
	Step 2: Second cup upright (1 pt)	**65.0**	45.0
	Normalized task score (of 40 pt)	**66.25**	51.25
Spoon Sort	Step 1: Sort spoons into target (1 pt/spoon)	16.0	**28.0**
	Normalized task score (of 25 pt)	16.0	**28.0**
Plate Placement	Step 1: Place plate in rack (0.5 pt)	**100.0**	96.0
	Step 2: Stack plate flush on the existing stack (0.5 pt)	**60.0**	**60.0**
	Normalized task score (of 25 pt)	**80.0**	78.0
Average normalized task score (weighted by rollout count)		48.18	**59.24**

ranges, with a fixed environment map providing ambient light. The domain-shift condition perturbs the following axes:

- **Lighting**: a secondary directional light source is added, with intensity and direction drawn from shifted distributions.
- **Environment map**: the ambient environment map is sampled from a discrete set of twenty maps unseen during training.
- **Camera parameters**: randomized deltas are applied on top of the nominal scene-camera extrinsics and intrinsics (and wrist-camera intrinsics, where applicable).
- **Textures and colors**: alternate textures and colors are sampled for the manipulands, distractors, and tabletop; the magnitude of the shift varies by scene.
- **Distractors**: random distractor objects are added in a subset of scenes.

All axes are resampled once per rollout from a deterministic initial-condition seed and held fixed for the duration of the rollout; the same 100 seeds per (task, condition) are shared across all policies, so every policy is evaluated under matched perturbations.

D.5 Statistical Significance

We evaluate each (policy, task, condition) cell with up to $n = 100$ independent rollouts (95–100 completed per cell on the LBM benchmark; App. B) and report mean success. Violins show the Beta posterior $\mathrm{Beta}(1+k, 1+n-k)$ over the true success rate, given k observed successes in n completed rollouts. Compact letter displays use a sequentialized Barnard test per task; the aggregated panel uses a Bonferroni-corrected Beta posterior test across the 13 tasks.

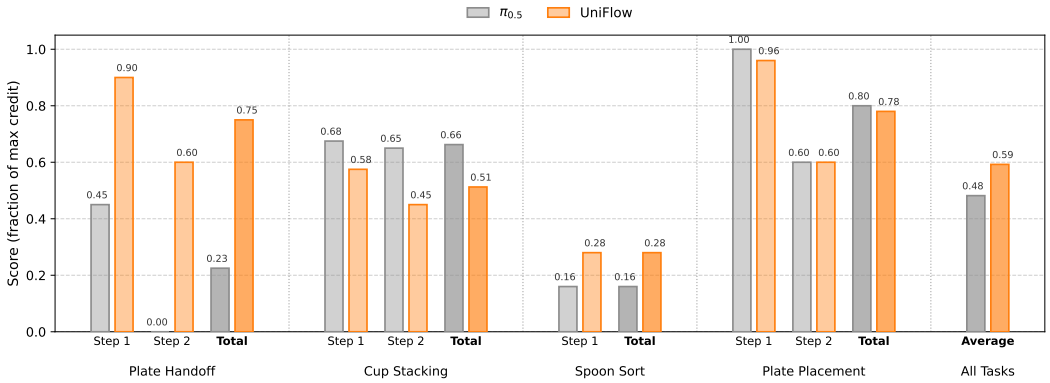


Figure 15: **Per-step breakdown of the real-world evaluation.** For each task, bars report the credit earned on each step as a fraction of that step's maximum credit, averaged over the evaluation rollouts (n=20 per task; n=25 for Plate Placement), for $\pi_{0.5}$ (gray) and UniFlow (orange). The **Total** column shows the normalized task score, i.e., total credit earned over maximum attainable credit, equal to the mean of the task's equally weighted step scores. Spoon Sort consists of a single step whose credit is pooled over the 25 spoons presented across its 20 rollouts, so its total coincides with Step 1. The rightmost pair (**Average**) reports the average normalized task score across the four tasks, weighted by rollout count (20+20+20+25=85 rollouts per policy). All plotted values are rounded to two decimals; exact values are given in Table 6, and the scoring rubric in App. D.2.

E Additional Results

E.1 Per-Modality Modulation Ablation

Table 7: Backbone ablation: per-task success rate (%) on three RoboTwin tasks, 100 rollouts per task. All variants share the same encoders, objective, action head, and training recipe; only the fusion backbone changes. **Bold** marks the best per column.

Backbone	Adjust Bottle	Open Laptop	Place Shoe	Avg
DiT-X	91.0	72.0	31.0	64.7
MMDiT	95.0	76.0	35.0	68.7
MMDiT-Any (UniFlow)	**99.0**	**82.0**	**42.0**	**74.3**

This ablation isolates the contribution of the MMDiT-Any design by comparing three fusion backbones that share identical encoders, objective, action head, and training recipe:

- **MMDiT-Any (UniFlow)**: each modality stream keeps its own AdaLN scale/shift, attention-projection, and feed-forward parameters, with a single joint self-attention over the concatenation of all streams.
- **MMDiT**: RGB and pointmap tokens are concatenated into a single observation stream sharing one set of modulation, attention-projection, and feed-forward parameters, as in the original two-stream MMDiT [9].
- **DiT-X**: cross-attention conditioning in the style of ManiFlow [2]; action tokens form the only transformer stream and attend to the fused observation tokens via cross-attention.

All variants are trained identically on three RoboTwin tasks (*adjust bottle*, *open laptop*, and *place shoe*) for 200 epochs, and evaluated over 100 randomized rollouts per task. As shown in Table 7, UniFlow outperforms both alternatives on every task, averaging 74.3% vs. 68.7% for MMDiT and 64.7% for DiT-X. The gap over MMDiT isolates per-modality parameterization: giving each modality its own modulation, attention-projection, and feed-forward parameters, rather than one set shared across the concatenated stream.

E.2 3D Encoder Ablation

Each ablation variant swaps *only* the geometry stream of UniFlow: the pointmap encoder (App. C.1) is replaced by PointNet++ [15], RandLA-Net [27], or Uni3D [19]; the RGB stream, backbone, objective, and action head are unchanged. The point-cloud encoders consume the same head-camera depth as the pointmap, back-projected to camera-frame XYZ, cropped to the workspace bounding box, and farthest-point-sampled to 512 XYZRGB points. Uni3D is initialized from its released pretrained checkpoint; RandLA-Net and PointNet++ are trained from scratch; all encoders are fine-tuned end-to-end (no weights are frozen). All variants share an identical recipe with no per-encoder tuning: single-task training on RoboTwin *Lift Pot* with 50 demonstrations for 100 epochs, AdamW (learning rate 10^{-4}, weight decay 10^{-3}) with a cosine schedule and 500 warmup steps, and EMA weights at evaluation. Success rates are over 100 randomized rollouts. With this recipe, UniFlow's pointmap stream reaches 83% success, compared with 42% for RandLA-Net, 22% for Uni3D, and 8% for PointNet++ (Table 8; Section 5.4).

Table 8: 3D encoder ablation on RoboTwin *Lift Pot*: success rate (%) over 100 rollouts when UniFlow's pointmap stream is replaced by a point-cloud encoder (512 XYZRGB points). **Bold**: best.

3D encoder	SR
PointNet++ [15], from scratch	8.0
Uni3D [19], pretrained	22.0
RandLA-Net [27], from scratch	42.0
UniFlow (pointmap + ConvNeXt V2)	**83.0**

E.3 Pretrained Point-Cloud Encoder and Capacity Ablations

Table 9 reports two further ablations on the RoboTwin tasks *Adjust Bottle* and *Lift Pot*, using only the head camera, 50 demonstrations per task, and 100 evaluation rollouts per task; the average is taken over the two tasks. These runs are separate from those of App. E.2 (e.g., UniFlow reaches 80% on *Lift Pot* here and 83% there).

Pretrained point-cloud encoder. We replace only UniFlow's 3D encoder with a point transformer pretrained with Sonata [20], and compare it with a from-scratch PointNet stream and with the ManiFlow baseline, which uses a from-scratch point-cloud encoder. For consistency and training time, 256 points are used for all point-cloud variants, which may differ from the optimal setting of each encoder. Together with the Uni3D result of App. E.2, this comparison indicates that, in our setting, pretrained point-cloud encoders do not close the gap to a pretrained 2D backbone applied to pointmaps.

Capacity and training budget. To provide a closer capacity comparison with ManiFlow, we train a smaller UniFlow whose fusion transformer has 4 layers of width 512 (instead of 8 layers of width 768), as well as the full-size model trained for half as many epochs (50 instead of 100). The small variant has fewer transformer parameters than ManiFlow's DiT-X but more parameters in total (outside the fusion transformer), so the comparison is closer to, but not strictly, parameter-matched.

E.4 Pointmap-Encoder Pretraining

To test whether 2D pretraining benefits the pointmap stream, we keep the architecture, data, and compute fixed and change only the initialization of the ConvNeXt V2 pointmap encoder. Initialized from pretrained weights, UniFlow achieves a success rate of 65.0% on the LBM task PUTREDBELLPEPPERINBIN, compared with 44.0% when the same encoder is randomly initialized. These numbers come from a separate experiment and are not comparable in absolute terms with Table 1 (e.g., 81.0% in-domain for UniFlow on this task); only the gap between the two initializations is meaningful.

Table 9: Additional RoboTwin ablations (head camera only, 50 demonstrations per task, 100 rollouts per task; Avg is over the two tasks). (a) 3D encoder; (b) model capacity and training budget. Transformer: parameters of the fusion transformer; Total: all parameters. **Bold** marks the best success rate per column.

(a) 3D encoder			
	Adjust Bottle	Lift Pot	Avg
ManiFlow (point cloud, from scratch)	77	31	54.0
UniFlow + Sonata (pretrained)	78	32	55.0
UniFlow + PointNet (from scratch)	77	43	60.0
UniFlow (pointmap + ConvNeXt V2)	**99**	**80**	**89.5**
(b) Capacity and training budget			
	Transformer (M)	Total (M)	Avg
ManiFlow (DiT-X, 8 layers, width 768)	118.1	175.4	54.0
UniFlow (MMDiT-Any, 4 layers, width 512)	56.7	237.9	77.0
UniFlow (MMDiT-Any, 8 layers, width 768), half training	255.0	440.0	81.0
UniFlow (MMDiT-Any, 8 layers, width 768), full training	255.0	440.0	**89.5**

E.5 Domain Shift by Axis

To separate appearance shifts from spatial shifts, we re-evaluate UniFlow on all 13 LBM tasks with groups of the domain-shift perturbations of App. D.4 applied in isolation: lighting, texture, and background (appearance) and camera parameters (spatial). Table 10 reports the average success rate over the 13 tasks for each axis; applying all axes together recovers the full domain-shift condition reported in Table 1. We target 100 episodes per task and axis; as in the main evaluation, a small number of trials were dropped due to occasional software crashes.

Table 10: Domain shift by axis: UniFlow's success rate (%) averaged over the 13 LBM tasks, with one group of domain-shift perturbations applied at a time. In-domain and Full shift are UniFlow's averages in Table 1.

In-domain	Lighting	Texture	Background	Camera	Full shift
62.3	62.4	61.6	59.9	61.5	60.5

Although camera-frame pointmaps change when a scene camera moves, UniFlow's average success rate drops by only 0.8 points under the camera axis. Characterizing robustness to larger viewpoint changes is left to future work.

F Real-world Qualitative Examples

We show qualitative rollout comparisons between UniFlow and the $\pi_{0.5}$ [11] baseline on each of the four YAM bimanual real-world tasks. Each filmstrip reads left-to-right in execution order; the top row is $\pi_{0.5}$ and the bottom row is UniFlow. The two rollouts are temporally aligned: frames in the same column correspond to the same elapsed time since the start of execution, as labeled below each column. This alignment allows direct visual comparison of task progress between the two policies at identical elapsed times.

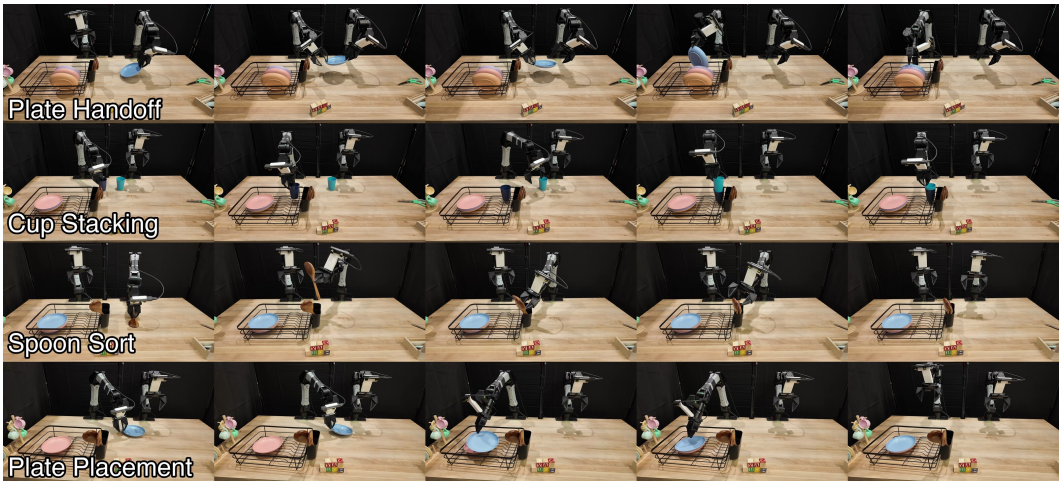


Figure 16: **Qualitative rollouts across tasks.** Each row shows a representative rollout for one task. From top to bottom: *Plate Handoff*, *Cup Stacking*, *Spoon Sort*, and *Plate Placement*. UniFlow executes diverse manipulation behaviors, including handing off, stacking, and placing objects, fusing 2D and 3D information from multi-view RGB-D observations.

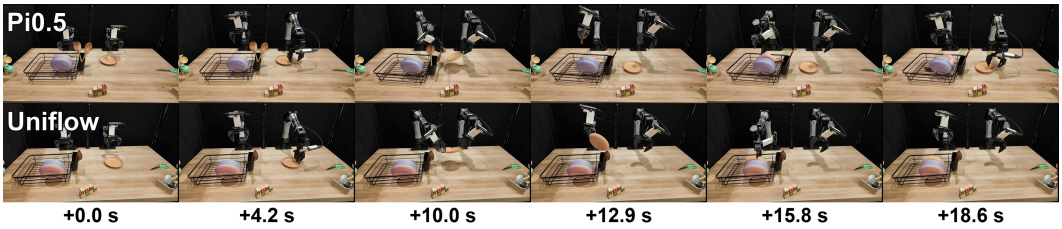


Figure 17: **Plate Handoff.** Execution comparison between $\pi_{0.5}$ (top) and UniFlow (bottom). The plate starts on the table; the policy must pick it up with one arm, hand it off to the other arm, and place it into the rack.

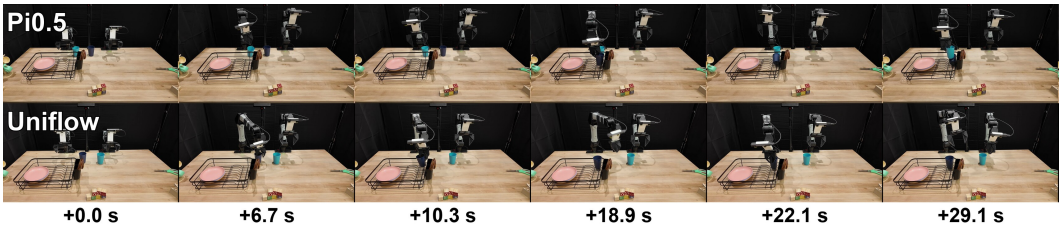


Figure 18: **Cup Stacking.** Execution comparison between $\pi_{0.5}$ (top) and UniFlow (bottom).

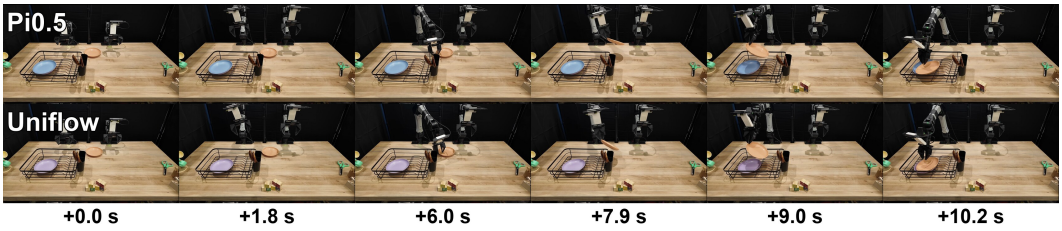


Figure 19: **Plate Placement.** Execution comparison between $\pi_{0.5}$ (top) and UniFlow (bottom).

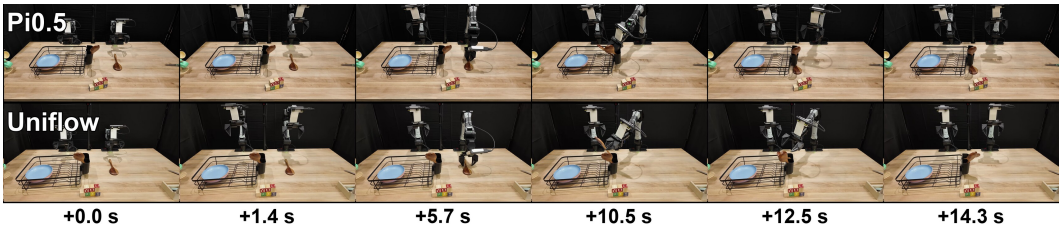


Figure 20: **Spoon Sort.** Execution comparison between $\pi_{0.5}$ (top) and UniFlow (bottom).

G Qualitative Rollouts on the LBM Benchmark

To complement the aggregate statistics of Tables 3 and 4, we visualize how the four policies behave on individual LBM benchmark rollouts. All frames are the simulator's head-camera view, and the same initial-condition (IC) index denotes the same randomized starting scene across policies, so every comparison is matched. We use two complementary views. *Final-frame success grids* (Figure 21) tile the terminal frame of one IC per task, shared by all four policies, into a 13×4 grid, bordered green for success and red for failure — an at-a-glance map of which policy clears which task on that IC. *Filmstrips* (Figures 22–31) unroll a single IC over time, sampling one frame every 5 s with the final frame bordered by outcome, exposing the temporal structure of a success or failure. In every filmstrip the rows are, top to bottom, LBM, UniFlow Image-only, UniFlow Pointmap-only, and the full UniFlow model; the domain-shift filmstrips use the ICs of the domain-shift grid, and the in-domain filmstrips use randomly chosen ICs. All panels show single rollouts and are illustrative; the success rates of Table 1 remain the quantitative comparison.

G.1 Final-Frame Success Grids

The grids illustrate the central finding: UniFlow is the only policy among the top two under *both* conditions. In-domain (Figure 21a) the image-only baseline is marginally the greenest column (9 of 13), with full UniFlow a close second (8) — the qualitative echo of its slight in-domain disadvantage in Table 1 (67.2 vs. 62.3). Under shift (Figure 21b) that advantage inverts: image-only fails on 10 of the 13 displayed ICs and LBM on 9, while UniFlow succeeds on all 13. This panel overstates the margin: in Table 1, UniFlow averages 60.5% under shift, and the pointmap-only baseline — geometry being largely lighting- and texture-invariant — partially keeps up (43.5%, vs. 25.9% for LBM and 14.4% for image-only).

G.2 Domain-Shift Rollouts

Under domain shift the filmstrips show UniFlow driving each task to completion while the baselines often stall, mis-grasp, or time out. Which baseline (if any) keeps pace is task-dependent: the pointmap-only stream succeeds on StackPlatesOnTableFromDryingRack (Figure 23) where the RGB-only stream fails, whereas RGB succeeds on PutRedBellPepperInBin (Figure 22) where pointmap fails, and LBM succeeds on PutRedBellPepperInBin and PutSpatulaInUtensilCrock (Figure 24). On StoreCerealBoxUnderShelf (Figure 25) and PutBananaOnSaucer (Figure 26) UniFlow is the *only* policy that succeeds in the displayed rollout. This mirrors the per-task statistics: under shift, no baseline is in the top statistical group on more than 7 of the 13 tasks, whereas UniFlow is on all 13 (Figure 12).

(a) In-domain (b) Domain shift

Figure 21: **Final-frame success grids on the LBM benchmark.** Each row is one of the 13 tasks (same order as Table 3) at one IC shared by all four policies; columns are LBM, UniFlow Image-only, UniFlow Pointmap-only, and full UniFlow. Every cell is the terminal frame of that rollout, bordered green for success and red for failure. **(a)** In-domain, the image-only baseline and UniFlow clear the most tasks (9 and 8 of 13), followed by LBM (7) and the pointmap-only baseline (6), the same ordering as the in-domain averages of Table 1 ($67.2/62.3/56.8/46.1$). **(b)** Under domain shift, UniFlow succeeds on all 13 displayed ICs and each baseline on 3–4. These displayed rollouts overstate UniFlow's margin: the average success rates under shift are $60.5/43.5/25.9/14.4$ for UniFlow, Pointmap-only, LBM, and Image-only.

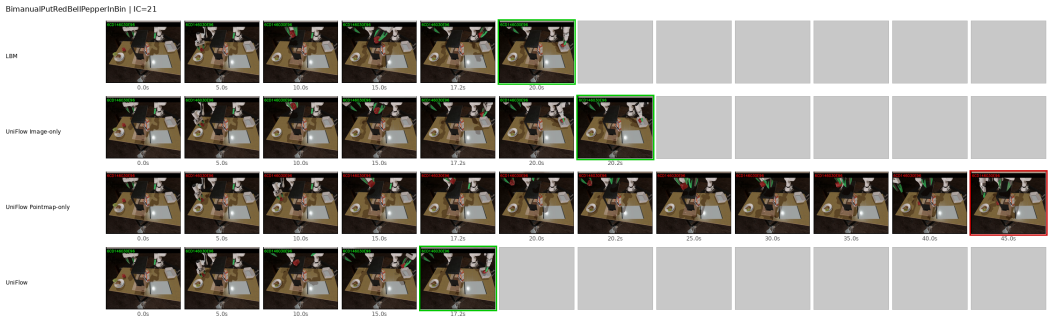


Figure 22: **Domain shift — PUTREDBELLPEPPERINBIN (IC 21).** LBM, Image-only, and UniFlow place the pepper; the pointmap-only ablation fails.

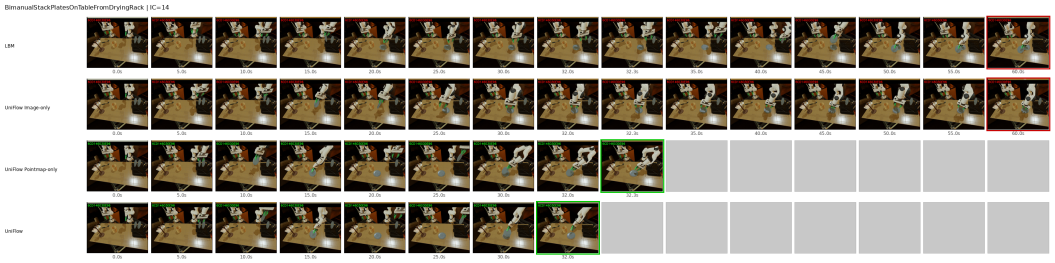


Figure 23: **Domain shift — STACKPLATESONTABLEFROMDRYINGRACK (IC 14).** The geometry-aware pointmap-only ablation and full UniFlow complete the stack; LBM and the image-only ablation fail under the shifted visual statistics.

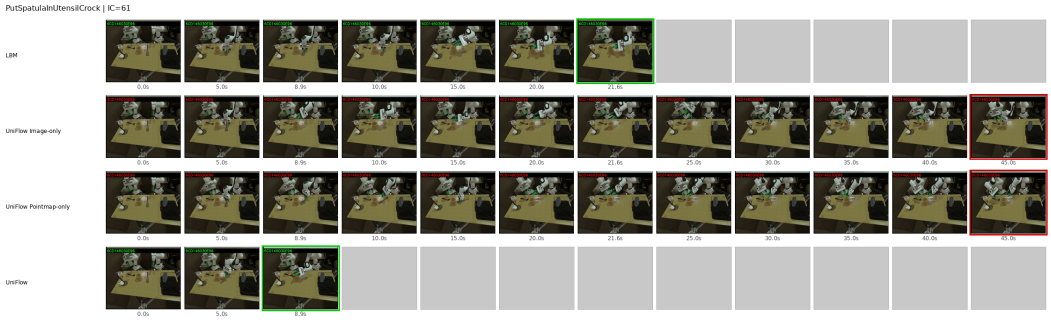


Figure 24: **Domain shift — PUTSPATULAINUTENSILCROCK (IC 61).** LBM and UniFlow succeed; both single-modality ablations fail.

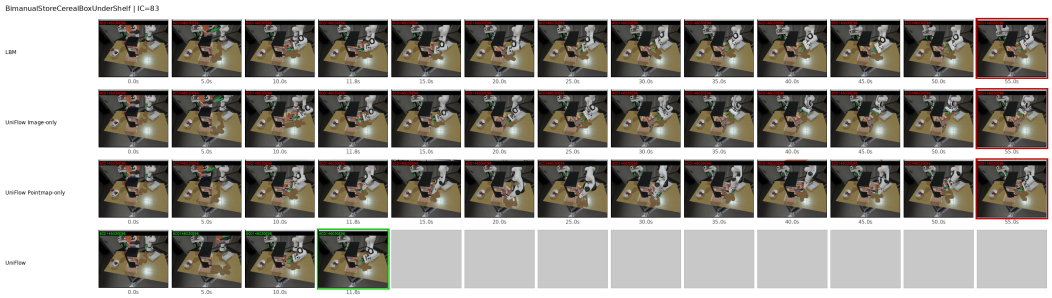


Figure 25: **Domain shift — StoreCerealBoxUnderShelf (IC 83).** Only full UniFlow completes the task; all three baselines fail.

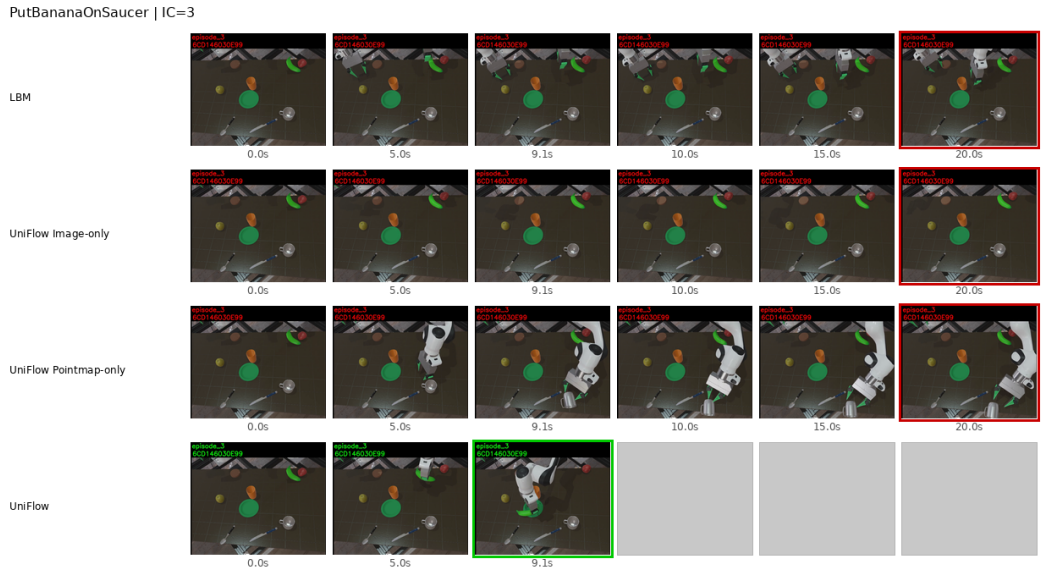


Figure 26: **Domain shift — PutBananaOnSaucer (IC 3).** A hard, low-success task: only UniFlow places the banana, consistent with its 19% vs. $\leq 3\%$ margin in Table 1.

G.3 In-Domain Rollouts

In-domain, where test-time RGB statistics match training, the image-only baseline and full UniFlow complete the task in most of these rollouts (Figures 27–30), with LBM and the pointmap-only baseline succeeding on some initial conditions as well. On PlaceFruitFromBowlIntoBin (Figure 31) the image-only baseline completes the task while UniFlow does not.

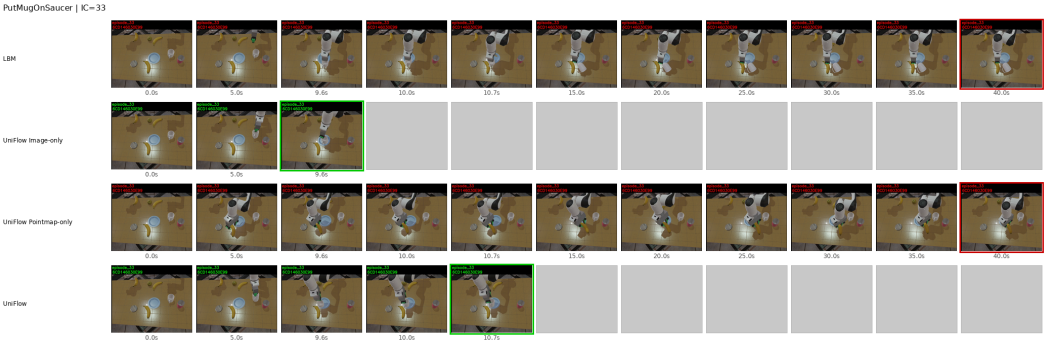


Figure 27: **In-domain — PUTMUGONSAUCER (IC 33).** Image-only and UniFlow place the mug; LBM and the pointmap-only ablation fail.

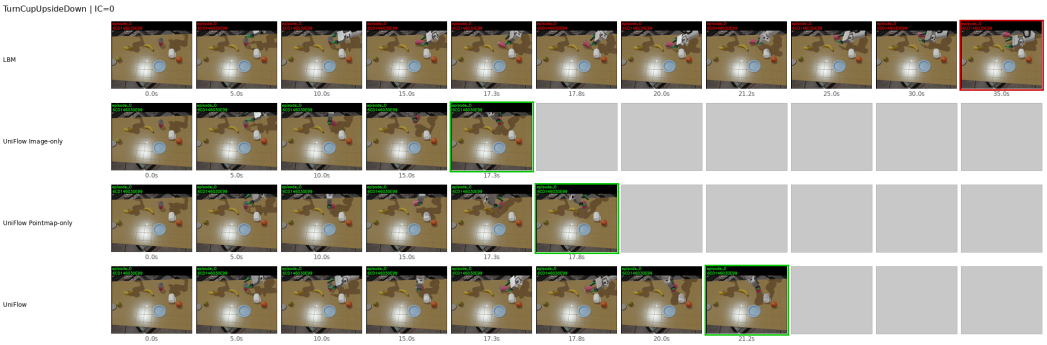


Figure 28: **In-domain — TURNCUPUPSIDEDOWN (IC 0).** Every policy except LBM completes the flip.

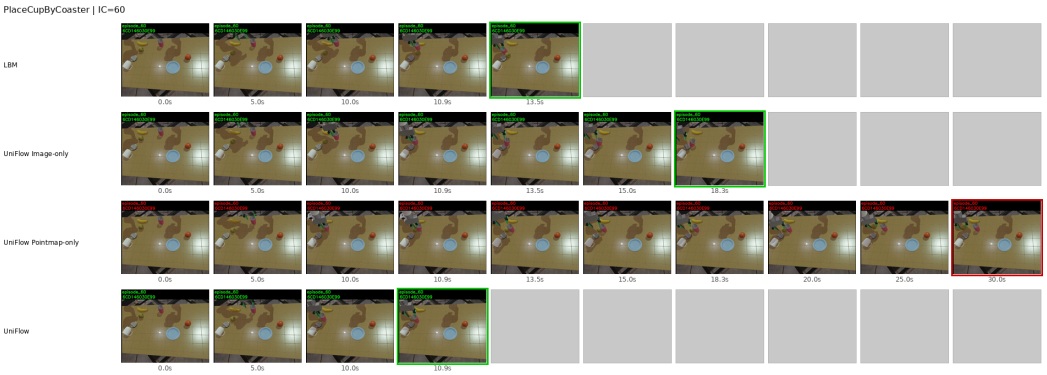


Figure 29: **In-domain — PLACECUPBYCOASTER (IC 60).** LBM, Image-only, and UniFlow succeed; the pointmap-only ablation fails on this low-success precision task.

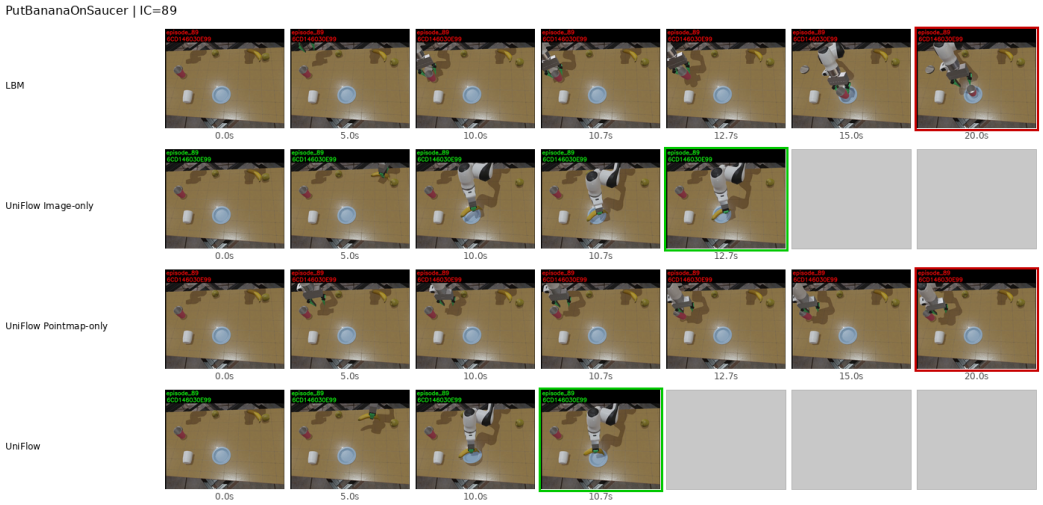


Figure 30: **In-domain — PUTBANANAONSAUCER (IC 89).** Image-only and UniFlow place the banana; LBM and pointmap-only fail.

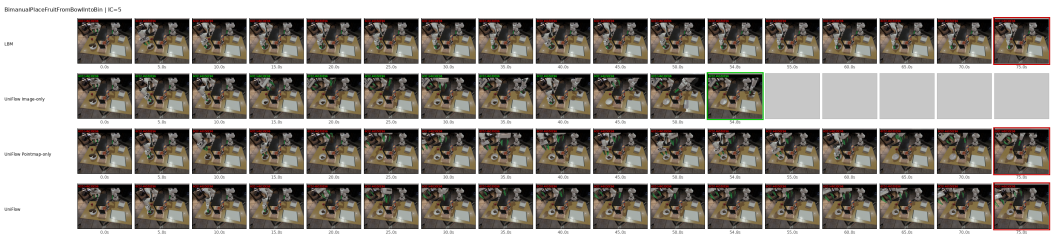


Figure 31: **In-domain failure case — PLACEFRUITFROMBOWLINTOBIN (IC 5).** The image-only baseline succeeds while full UniFlow fails (LBM and pointmap-only also fail) — the qualitative counterpart to image-only's slight in-domain edge in Table 1.

H Compute and Reproducibility

On the LBM benchmark, LBM is trained on 4 nodes for 200k gradient steps and UniFlow on 4 nodes for 180k gradient steps. On RoboTwin, all models in Table 2a are trained on 8 NVIDIA A100 GPUs for 500 epochs; the RoboTwin ablations use shorter schedules (200 epochs in App. E.1; 100 epochs in App. E.2 and for the full-size model in App. E.3). Real-world models are each trained on two NVIDIA H200 GPUs for 40k gradient steps; this applies to both $\pi_{0.5}$ and UniFlow.

For reproducibility, every (policy, task, condition) cell is evaluated with 100 rollouts in simulation (on the LBM benchmark, 95–100 completed per cell; App. B) and 20 rollouts in the real world (25 for Plate Placement; see App. D.2). Code will be released upon publication.