BECKMANN WORLD MODELS: EFFICIENT LEARNING OF ONE-STEP DYNAMICS

Anonymous authors
Paper under double-blind review

ABSTRACT

A one-step world model must predict the consequences of an action while remaining inexpensive to learn. DriftWorld achieves one-call prediction by training with a population of generated futures for each condition. We study a different construction through Beckmann Transport Models, which learn a terminal map from local noise–data pairs and an input derivative. The map directly predicts an action-conditioned future. We adapt it through direct supervision of the pure-noise endpoint and motion-aware losses, while retaining one-call inference. On Robomimic Can, the resulting model reduces 64-frame LPIPS from 0.0682 to 0.0090, moving-region MSE from 0.0769 to 0.0316, and full-episode MSE from 0.0256 to 0.0112 relative to our trained DriftWorld baseline. It reaches this result in 4.32 cumulative training-loop hours, including all initialization training, against 10.19 hours for DriftWorld. At a matched batch size, the complete objective achieves $3.33\times$ higher training throughput with 7.77 GiB peak allocated memory against 36.07 GiB. PushT reveals a diffusion quality–latency tradeoff and separates prediction from decision quality. These results establish conditional terminal-map learning as an effective route to accurate one-step prediction with lower measured training cost.

1 INTRODUCTION

Action-conditioned world models predict the visual consequences of proposed controls. Planning requires many candidate futures. Diffusion world models generate them through iterative denoising (Yang et al., 2024, Qi et al., 2026), whereas DriftWorld predicts a future chunk in one network evaluation (Lu et al., 2026a). DriftWorld trains with a population of generated futures for each condition. This work targets accurate one-step prediction with lower training cost.

Beckmann Transport Models (BTM) provide a promising starting point (Lee et al., 2026). They learn the terminal map of an autonomous transport. In a world model, its destination is a conditional future video chunk. The direct-map objective uses local noise–data pairs and a Jacobian–vector product (JVP), without constructing a generated population for each condition. It also makes the final RGB prediction directly available for supervision. These properties offer a route to efficient learning without a pretrained diffusion teacher.

A direct port nevertheless performs poorly. The initial PushT implementation reaches LPIPS 0.8087, and even a stabilized map remains weaker than DriftWorld. Two gaps motivate the adaptation. First, preserving clean data constrains the map at recorded futures, while deployment starts from pure noise. Second, an image-wide loss can underweight the small regions that carry robot and object motion. Beckmann World Models address these gaps through *source anchoring*, which supervises the deployed noise-to-future map, and *motion-aware endpoint supervision*, which emphasizes spatial and temporal changes. The transport derivative holds history and actions fixed, and short generated-history recovery supports the same endpoint. Inference remains one call per native chunk.

The completed Robomimic Can recipe improves both short- and long-horizon prediction over task-matched DriftWorld (Figure 1). Including all parent training, it takes 4.32 recorded training-loop hours against 10.19 hours. Complete-update profiling measures $3.33\times$ higher throughput. The compared recipes use different optimizers and supervision.

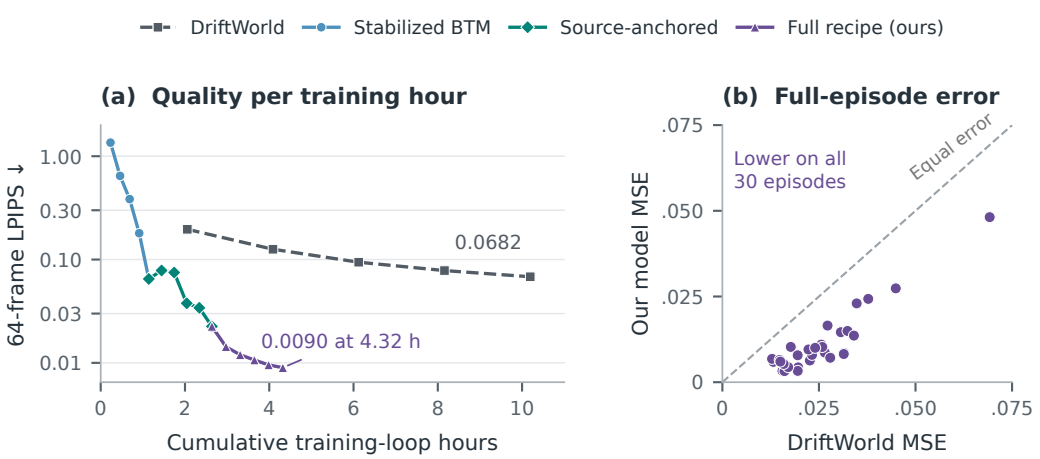

Figure 1: **Accurate futures with lower learning cost on Robomimic Can.** (a) Validation LPIPS against cumulative training-loop hours, including all parents. Colors identify successive recipes and markers denote evaluated checkpoints. Both methods use two A800 GPUs, with different batches and optimizers. (b) Full-episode raw MSE of the final models on all 30 validation episodes. Every point lies below equal error.

The contributions are threefold:

- A conditional terminal-map formulation that exposes the deployed prediction to source and motion supervision while preserving one-call inference.
- An empirical study of the adaptation from an initial BTM port to an effective world model, including endpoint diagnostics, matched qualitative comparisons, and complete training-cost measurements.
- An evaluation of prediction accuracy, computational efficiency, and decision quality, with a released diffusion baseline and separate PushT planning and policy-ranking tests.

The transport theory and backbone are inherited. The contribution is their adaptation and evaluation for conditional visual dynamics.

2 Related Work

Action-conditioned world models. UniSim, IRASim, and Ctrl-World use generative video prediction to simulate interactions (Yang et al., 2024, Zhu et al., 2025, Guo et al., 2026). GPC and WorldGym evaluate action selection and policy quality (Qi et al., 2026, Quevedo et al., 2025). DriftWorld adapts Drifting to conditional video and studies prediction, control, and offline policy evaluation (Lu et al., 2026a). We retain its task interfaces and U-Nets to study terminal-map learning.

Fast diffusion and flow models. Diffusion and flow matching typically integrate learned generative dynamics (Ho et al., 2020, Lipman et al., 2023). Progressive distillation, adversarial distillation, and DMD2 accelerate pretrained teachers (Salimans & Ho, 2022, Sauer et al., 2024, Yin et al., 2024). Consistency models map trajectory states to a common endpoint (Song et al., 2023). Rectified flow, flow maps, and shortcut models provide additional routes to few-step generation (Liu et al., 2023, Boffi et al., 2025, Frans et al., 2025). MeanFlow learns average velocities (Geng et al., 2025), improved MeanFlow revises their regression objective (Geng et al., 2026), and subsequent work studies training dynamics and pixel-space prediction (Kim et al., 2026, Lu et al., 2026b). These works establish one-step inference, teacher-free learning, and JVPs. Our distinction concerns the conditional terminal map and its learning cost.

Drifting, Beckmann maps, and endpoint supervision. Drifting learns through distribution-dependent interactions among generated and data samples (Deng et al., 2026). BTM derives autonomous transports and their terminal maps under explicit support and regularity assumptions (Lee et al., 2026). Discrete BTM extends the construction to simplex-valued sequences for language modeling and reasoning (Tang & Wang, 2026). Our continuous video map conditions transport on a fixed action problem and adds direct supervision at the source endpoint. Motion losses emphasize local changes in that prediction. Generated-history supervision, used in Self Forcing for causal diffusion (Huang et al., 2025), addresses a separate context mismatch. We use a short recorded-trajectory recovery regularizer without adopting its diffusion or distribution-matching procedure.

3 METHOD

3.1 CONDITIONAL FUTURE PREDICTION

Let $h = (o_{-H+1}, \dots, o_0)$ be observed RGB history, $a = (a_0, \dots, a_{F-1})$ proposed controls, and $y = (o_1, \dots, o_F)$ the recorded future. With condition $c = (h, a)$, the model returns

$$\widehat{y} = T_\theta(z, c) = z + U_\theta(z, c), \qquad z \sim \mathcal{N}(0, 1.35^2 I). \tag{1}$$

Images are normalized to $[-1, 1]$. We use four history frames at 96×96 resolution and predict $F = 4$ frames on PushT or $F = 2$ on Robomimic Can. The residual U-Net is inherited from the task-matched DriftWorld baseline. Its history and action interface is unchanged. The map has no interpolation-time input. We write $\|v\|_D^2 = D^{-1}\|v\|_2^2$, with $D = 3F96^2$.

3.2 LEARNING A BECKMANN TERMINAL MAP

For fixed c, an autonomous field b_c generates trajectories satisfying $\dot{X}_\tau = b_c(X_\tau)$. For a target supported on a lower-dimensional data manifold, and under the regularity assumptions of Lee et al. (2026), the terminal map sends each point on a trajectory to the same destination and fixes the target support. Thus

$$T_c(X_\tau(x)) = T_c(x), \qquad J_x T_c(x) b_c(x) = 0, \qquad T_c(y) = y. \tag{2}$$

Figure 2a illustrates this geometry. Transport time τ is distinct from physical video time. These population identities motivate direct map learning, without estimating or integrating the field at inference. They are not convergence guarantees for the trained video model.

Sample $s \sim \mathcal{U}[0, 1]$ and form $x_s = (1 - s)z + sy$ and $d = y - z$. The stopped target is

$$p = T_\theta(x_s, c), \qquad q = J_x T_\theta(x_s, c)d, \qquad u = \mathrm{sg}(p + \eta q), \quad \eta = 0.25. \tag{3}$$

The derivative acts only on future RGB coordinates. History and actions remain fixed. The straight sampled chord is distinct from an autonomous trajectory, and Equation 2 does not assert that every sampled q vanishes. With $e_i = \|p_i - u_i\|_D^2$, training uses

$$\mathcal{L}_{\mathrm{tr}} = \frac{1}{B}\sum_{i=1}^{B} \frac{e_i}{\mathrm{sg}(e_i) + 0.01}, \qquad \mathcal{L}_{\mathrm{bd}} = \mathbb{E}\|T_\theta(y, c) - y\|_D^2, \tag{4}$$

$$\mathcal{L}_{\mathrm{near}} = \mathbb{E}\big[\mathbf{1}_{s>0.8}\|p - y\|_D^2\big], \qquad \mathcal{L}_{\mathrm{BTM}} = \lambda\mathcal{L}_{\mathrm{tr}} + \mathcal{L}_{\mathrm{bd}} + \mathcal{L}_{\mathrm{near}}. \tag{5}$$

The near-data expectation averages over the full batch. The detached λ balances output-layer gradient norms. The JVP constructs a target without a reverse-mode graph. Appendix A specifies the implementation and optimization choices.

3.3 ANCHORING THE SOURCE ENDPOINT

The clean boundary constrains $T_\theta(y, c)$, while deployment queries $T_\theta(z, c)$. We supervise this exact operation using

$$p_0 = T_\theta(z, c), \qquad \mathcal{L}_{\mathrm{src}} = \mathbb{E}\|p_0 - y\|_D^2. \tag{6}$$

Source anchoring uses the same noise draw as the interpolant and lies outside the transport gradient balance. Its coefficient ramps to one over 500 continuation updates. It adds a training pass but leaves inference unchanged.

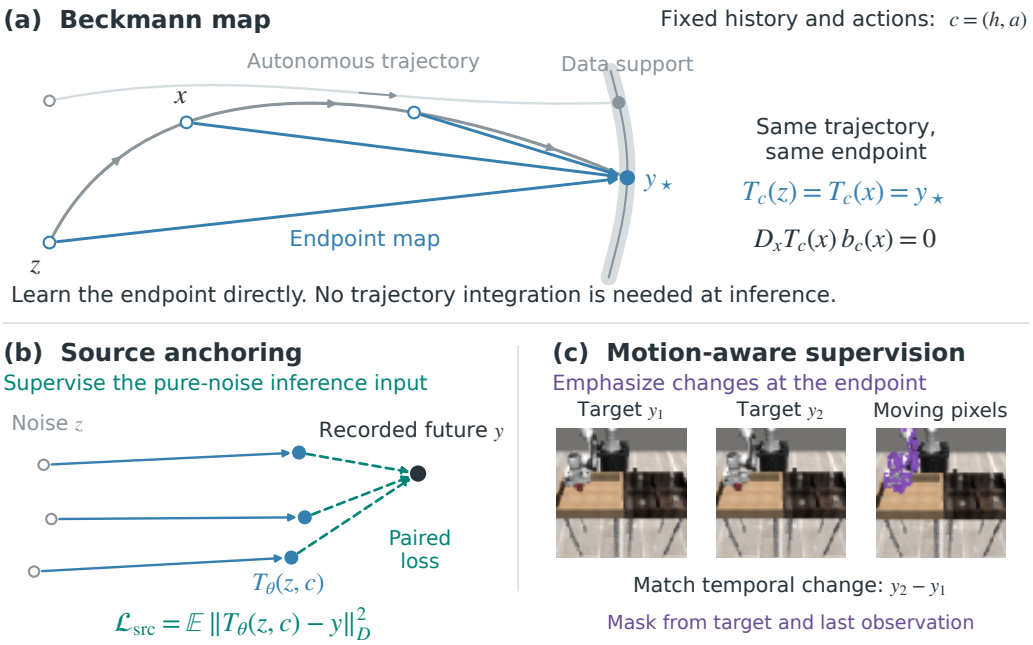

Figure 2: **From terminal transport to conditional prediction.** (a) Points on an autonomous trajectory share one destination. Blue arrows are direct map evaluations with fixed history and actions. (b) Source anchoring supervises the pure-noise deployment input. Dashed green arrows denote paired losses. (c) Recorded Robomimic Can targets define moving pixels and temporal differences. Panels (a) and (b) are conceptual, not measured video-space trajectories.

Independent paired squared error decomposes into conditional-mean error, source-induced output variance, and irreducible target variance (Appendix A). Source anchoring therefore rewards accurate conditional means but can suppress legitimate diversity. Reconstruction gains do not establish calibrated conditional sampling.

3.4 Motion-Aware Endpoint Supervision

Motion weighting also appears in DriftWorld (Lu et al., 2026a). Here, spatial and temporal losses act directly on the source endpoint:

$$\mathcal{A}(p, y) = 0.1\mathcal{L}_{\text{move}}(p, y) + 0.05\mathcal{L}_{\text{temp}}(p, y) + 0.01\mathcal{L}_{\text{range}}(p). \tag{7}$$

The moving-region loss uses a target-defined mask and a denominator floor that bounds sparse-region amplification. The temporal loss matches consecutive differences within the chunk. The range loss penalizes excursions outside normalized RGB bounds. Predictions remain unclipped during inference and autoregressive feedback.

A supporting branch predicts a recorded endpoint from a short, detached generated history every fourth update, with $\mathcal{L}_{\text{rec}} = \|p_{\text{rec}} - y_{\text{rec}}\|_D^2$. The complete objective is

$$\mathcal{L} = \mathcal{L}_{\text{BTM}} + \mathcal{L}_{\text{src}} + r_k\big[\mathcal{A}(p_0, y) + 0.25 I_k\{\mathcal{L}_{\text{rec}} + \mathcal{A}(p_{\text{rec}}, y_{\text{rec}})\}\big], \tag{8}$$

where $r_k = \min(k/1000, 1)$ and $I_k = \mathbf{1}_{(k+1) \bmod 4=0}$. Only the final recovery prediction receives gradients. This regularizes the endpoint under imperfect context using recorded targets. Algorithm 1 gives one complete update. Appendix A specifies the masks and prefix schedule. The complete recipe also reduces learning rates and adds training updates. These choices are evaluated jointly, without attributing their combined gain to the motion loss alone. All training paths use shared endpoint weights. At deployment, Equation 1 remains the entire sampling operation.

Algorithm 1 Conditional endpoint learning

Require: Recorded window (h, a, y), valid recovery sequence, parameters θ, update count k
$c \leftarrow (h, a)$, $z \sim \mathcal{N}(0, 1.35^2 I)$, $s \sim \mathcal{U}[0, 1]$
$x_s \leftarrow (1 - s)z + sy$, $d \leftarrow y - z$
$p \leftarrow T_\theta(x_s, c)$, $u \leftarrow \mathrm{sg}\left(p + 0.25 J_x T_\theta(x_s, c) d\right)$ ▷ Hold c fixed
Compute $\mathcal{L}_{\mathrm{BTM}}$ from transport, clean-boundary, and near-data losses
$p_0 \leftarrow T_\theta(z, c)$, $\mathcal{L} \leftarrow \mathcal{L}_{\mathrm{BTM}} + \|p_0 - y\|_D^2 + r_k \mathcal{A}(p_0, y)$
if $(k + 1) \bmod 4 = 0$ **then**
    Generate a raw prefix with recorded actions and no reverse-mode graph
    Read target y_{rec} and form c_{rec} from generated history and aligned actions
    $z_{\mathrm{rec}} \sim \mathcal{N}(0, 1.35^2 I)$, $p_{\mathrm{rec}} \leftarrow T_\theta(z_{\mathrm{rec}}, c_{\mathrm{rec}})$
    $\mathcal{L} \leftarrow \mathcal{L} + 0.25 r_k [\|p_{\mathrm{rec}} - y_{\mathrm{rec}}\|_D^2 + \mathcal{A}(p_{\mathrm{rec}}, y_{\mathrm{rec}})]$
end if
Backpropagate $\mathcal{L}$, clip gradients, update θ, scheduler, and EMA
Ensure: Deployment returns $T_{\mathrm{EMA}}(z, h, a)$ in one network evaluation

4 Experiments

4.1 Experimental Setup

Tasks and baselines. PushT and Robomimic Can use the released DriftWorld RGB interfaces and inherited backbones, with approximately 8.4 and 74.2 million parameters (Lu et al., 2026a, Mandlekar et al., 2021). The primary baseline is DriftWorld trained on the same split. A released GPC diffusion checkpoint is also evaluated on our PushT cohort, with its distinct training provenance disclosed. Published GPC, AVDC, Ctrl-World, and MSE U-Net scores provide additional context in Appendix E.

Evaluation protocol. PushT has 3,608 training and 400 validation episodes, of which 388 support 64-frame evaluation. Robomimic Can has 270 training and 30 validation episodes. Episodes are split before window construction. Validation sets were used during development. The 64-frame benchmark starts from four observed frames and scores 60 generated frames, requiring 15 map calls on PushT and 30 on Robomimic Can. Full episodes test longer use. Raw MSE, target-defined moving-region MSE, SSIM, and LPIPS are averaged within episodes, then equally across episodes (Wang et al., 2004, Zhang et al., 2018). Appendix C specifies preprocessing. Tables use final checkpoints of completed recipes and count all inherited training. The unfinished PushT motion-aware recipe appears only in Appendix B.6.

4.2 Prediction Accuracy

Table 1: **Prediction results.** Four observed images initialize 60 generated frames. Locally trained rows share the task-specific backbone and split but differ in training budget. Released GPC is reevaluated on the same PushT cohort with different training provenance. Boldface compares locally trained methods only. The complete recipe is available on Robomimic Can. Dashes denote unmeasured full-episode scores.

Task	Method	64 total frames				Full episode
		LPIPS ↓	Moving MSE ↓	MSE ↓	SSIM ↑	MSE ↓
Robomimic Can	DriftWorld	.0682	.0769	.0104	.856	.0256
	Source-anchored	.0225	.0576	.0068	.939	.0940
	Ours	**.0090**	**.0316**	**.0031**	**.974**	**.0112**
PushT	DriftWorld	.0790	.4093	.0314	.927	.0451
	Source-anchored	.0746	.3564	.0293	.942	.0434
	Learned direction	**.0501**	**.2632**	**.0208**	**.951**	**.0319**
PushT	GPC (released)	.0324	.2446	.0192	.966	—

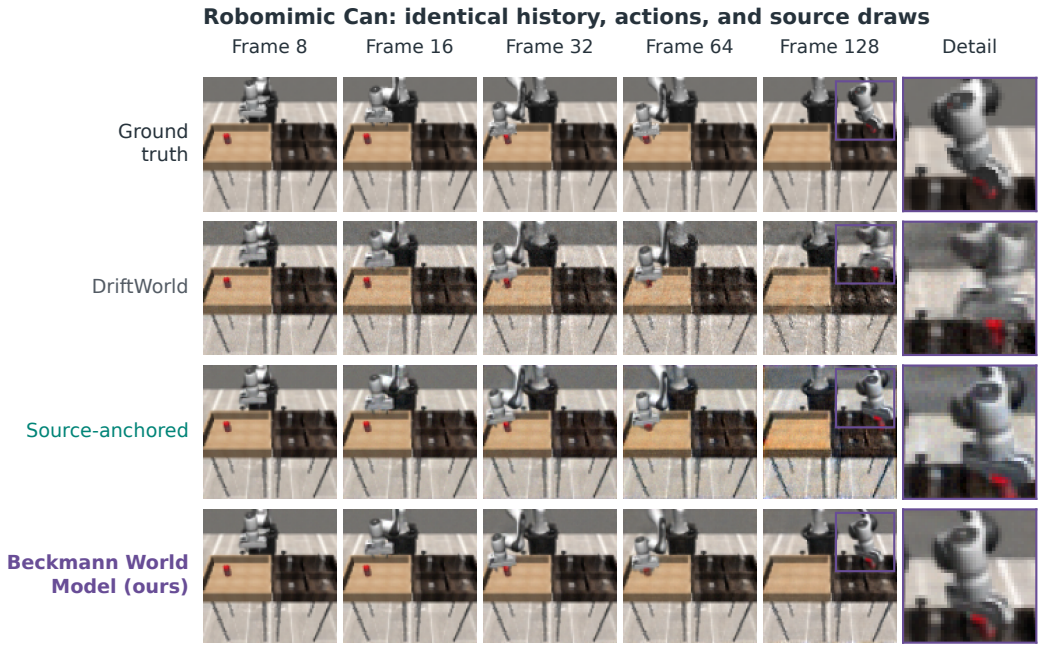

Figure 3: **Matched Robomimic Can rollouts.** The first validation episode, fixed before inference, with shared real history, recorded actions, and standardized source draws. The last column enlarges the outlined region at frame 128. The full recipe retains more of the recorded scene than source anchoring in this example. Display clipping is not used for feedback or MSE. Appendix D includes three predeclared episodes and their complete error traces.

Autoregressive prediction. On Robomimic Can, the full recipe reduces 64-frame LPIPS by 86.8% and moving-region MSE by 58.9% relative to DriftWorld (Table 1). Paired episode-bootstrap intervals for these differences are $[-0.0615, -0.0568]$ and $[-0.0529, -0.0378]$. They describe evaluation episodes for fixed models, not training-run variation. Full-episode MSE falls from 0.0256 to 0.0112 and is lower on all 30 episodes. This resolves the source-anchored model's long-episode weakness despite its good 64-frame scores. Figure 3 shows matched rollouts from the completed checkpoints.

Released diffusion baseline. The released GPC checkpoint reaches LPIPS 0.0324 and MSE 0.0192 on the same 388 PushT inputs, better than the learned-direction model's 0.0501 and 0.0208. The learned-direction model trades higher prediction error for one evaluation per four frames, against 12 GPC denoiser evaluations. Appendix E.1 reports native inference timing and three fixed qualitative comparisons. GPC's training exclusion of this cohort is not established, so this comparison does not measure equal-data generalization.

Endpoint-map behavior. Figure 4 independently queries five interpolated inputs on 90 Robomimic Can windows with four shared source draws per window. The stabilized map has clean-boundary MSE 0.000105 but source-endpoint MSE 0.002831. Source anchoring lowers the latter to 0.000823, and the full recipe to 0.000503. A small boundary error therefore leaves a substantial deployment gap. Source-induced variance also falls, consistent with the source-loss decomposition (Appendix B.7). A separate one-call probe improves pixel, perceptual, and moving-region accuracy over DriftWorld and copy-last prediction (Table 8).

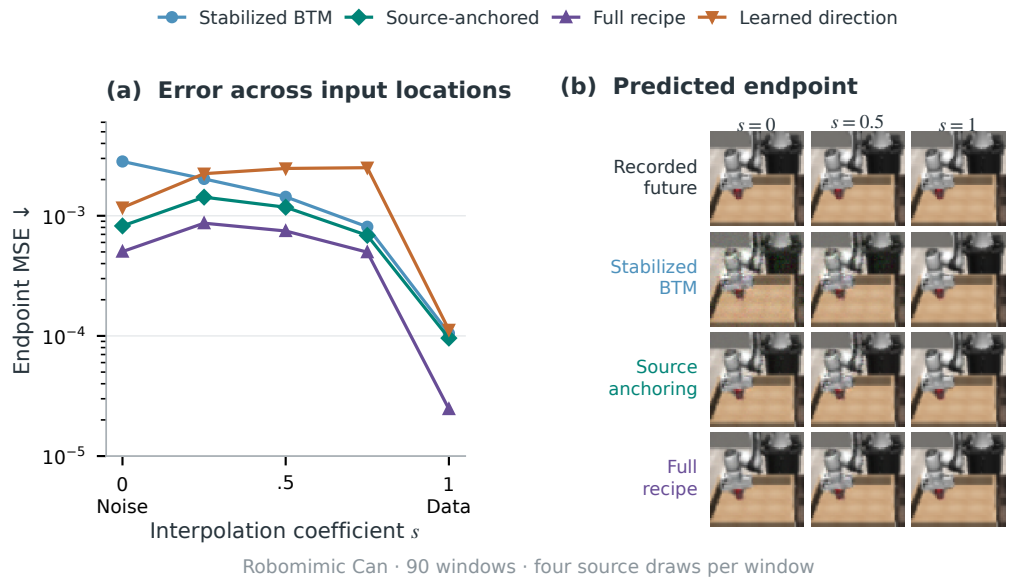

Figure 4: **Boundary accuracy leaves a source-endpoint gap.** (a) Raw MSE from independent one-call queries of $x_s = (1-s)z + sy$ on 90 Robomimic Can windows with four source draws each. (b) A fixed crop of the first episode's middle window, with the recorded future above. Only $s = 0$ is deployment. Intermediate inputs contain target information and do not constitute an inference trajectory.

Table 2: **Cumulative ablation from the initial BTM implementation.** Every row predicts a chunk in one call. Updates include inherited training. Stabilization combines normalization, gradient balancing, a near-data anchor, and optimizer changes. Motion-aware and learned-direction recipes are separate continuations from source anchoring. These are recipe comparisons, not isolated equal-budget loss ablations. Missing completed measurements are marked by a dash.

	PushT			Robomimic Can		
Training recipe	Updates (k)	64-frame LPIPS ↓	Full-episode MSE ↓	Updates (k)	64-frame LPIPS ↓	Full-episode MSE ↓
Initial BTM port	114.1	.8087	—	—	—	—
Stabilized map	141.7	.1119	.0627	5.0	.0649	.1159
+ source anchoring	163.3	.0746	.0434	10.0	.0225	.0940
+ motion-aware recipe	—	—	—	15.0	**.0090**	**.0112**
+ learned direction	183.0	**.0501**	**.0319**	15.0	.0854	.0759
DriftWorld reference	39.4	.0790	.0451	20.0	.0682	.0256

Cumulative component study. The initial PushT BTM port reaches LPIPS 0.8087. Stabilization lowers it to 0.1119, source anchoring to 0.0746, and the learned-direction continuation to 0.0501 (Table 2). Learned directions instead worsen Robomimic Can LPIPS to 0.0854. The motion-aware recipe yields the strongest Robomimic Can results. These continuations change training duration and, for the full recipe, learning rates. They compare complete recipes, not isolated causal effects of individual losses. A connected-recovery alternative and its additional optimization differences are reported in Appendix B.

4.3 Computational Efficiency

Cumulative training cost. Figure 1a counts all Robomimic Can parent training: 1.14 hours for the stabilized map, 1.50 for source anchoring, and 1.68 for the full recipe. The total is 4.32 hours against DriftWorld's 10.19 hours on two A800 GPUs, a 57.6% reduction at better final quality. Within a

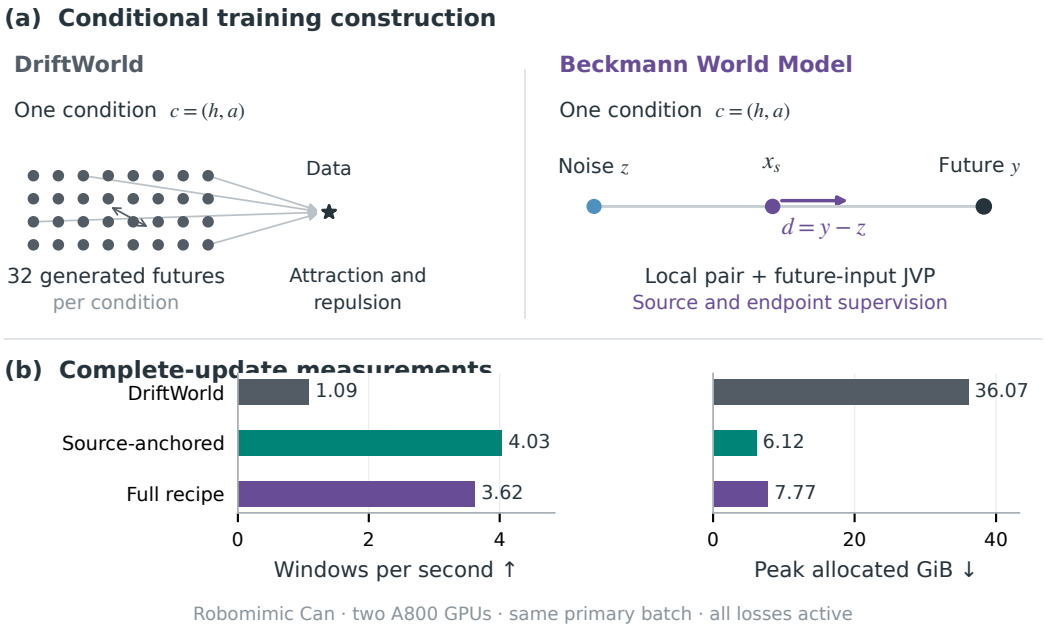

Figure 5: **Training construction and complete-update cost.** (a) DriftWorld's generated population and Beckmann's local noise–data pair, shown conceptually for one fixed condition. (b) Measurements on two A800 GPUs at one primary window per GPU, including the full recovery schedule. The full recipe retains $3.33\times$ higher throughput. Different losses and optimizers prevent isolating population size as the sole cause.

five-hour budget, the last evaluated checkpoints reach LPIPS 0.0090 at 4.32 hours and 0.1257 at 4.09 hours, respectively. No interpolation is used. Training-loop time excludes evaluation waits, interruptions, and data preparation. PushT's longer inherited training does not support an analogous lower-budget claim.

Complete-update cost. DriftWorld generates 32 futures per Robomimic Can condition and 16 per PushT condition. Local-pair learning replaces these populations with a JVP target and endpoint losses (Figure 5). With all losses active at the same primary batch size, the full recipe processes 3.62 windows per second against 1.09 and allocates 7.77 GiB against 36.07 GiB. The $3.33\times$ throughput advantage includes all model passes, derivatives, backward propagation, synchronization, optimization, and EMA. It is an implementation comparison with different optimizers, not an isolated group-size effect. Appendix B.3 specifies the complete profile.

Inference cost. For four PushT frames, the learned-direction map takes 23.8 ms against 138.8 ms for GPC on one A800 GPU. At batch size 16, complete-query times are 197.5 and 527.0 ms. DriftWorld has comparable one-call latency (Figure 13). The diffusion comparison therefore exposes a quality–latency tradeoff, while the advantage over task-matched DriftWorld concerns prediction and measured training cost.

4.4 Decision Quality

Action selection. Native PushT GPC-RANK scores 50 proposals from a frozen diffusion policy over 15 predicted steps, then executes eight actions. The base policy reaches 0.635 IoU. DriftWorld reaches 0.709, source anchoring 0.702, and learned direction 0.664. Paired intervals include zero for differences between these final models. Improving LPIPS along the learned-direction run does not improve its final action-selection mean (Figure 6a).

Offline policy evaluation. Across seven policies with 300 trials each, learned direction reaches Pearson correlation 0.9743 and Spearman correlation 0.9286, against 0.8565 and 0.7143 for local

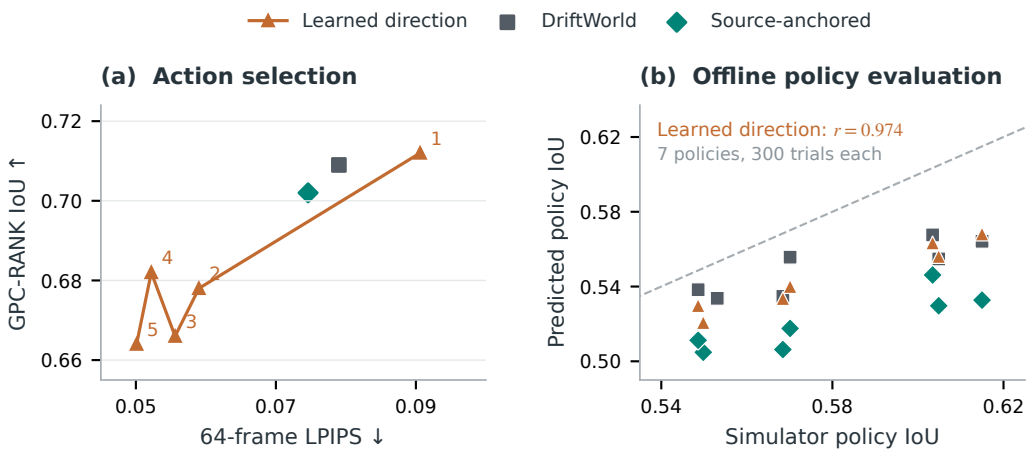

Figure 6: **Action selection and policy evaluation on PushT.** (a) Numbered points are successive learned-direction epochs. Improved LPIPS does not improve final GPC-RANK IoU. (b) Each point averages 300 trials for one policy. Learned direction tracks policy ordering with Pearson $r = 0.974$ while underpredicting absolute performance. Colors and markers are shared across panels.

DriftWorld (Figure 6b). Its absolute IoU gap remains larger, 0.0355 against 0.0306, with systematic underprediction. The released DriftWorld control has lower Pearson correlation 0.9255 but better rank correlation 0.9643 and absolute gap 0.0203. These descriptive results distinguish policy ordering, calibration, and action selection. Appendix C.5 gives individual scores and reference differences.

5 Discussion and Limitations

The strongest learning-cost evidence concerns Robomimic Can. Released GPC remains more accurate on PushT at greater inference cost. PushT has a longer inherited training history, and the validation splits were used during development. The cumulative recipes do not isolate individual losses or transport's marginal contribution. A reconstruction-only model trained from scratch under matched supervision and computation remains an important attribution control. Source anchoring may suppress valid uncertainty. Neither lower image error nor higher policy correlation establishes superior action selection or calibrated physical simulation.

6 Conclusion

Beckmann terminal maps represent action-conditioned futures directly. Source anchoring and endpoint supervision make the conditional map effective while preserving one-call prediction. On Robomimic Can, the completed recipe improves accuracy over task-matched DriftWorld with lower cumulative training time and higher update throughput. Diffusion-baseline evaluation and separate decision tests characterize the resulting quality, cost, and control tradeoffs.

Reproducibility Statement

The appendices specify conditioning, derivative semantics, loss coefficients, training lineage, metric preprocessing, bootstrap units, and timing scope. Qualitative examples use three validation positions fixed before inference. Complete-update profiling covers every mature recovery depth. Reported results come from model evaluations and recorded training intervals. The study does not include independent training repetitions.

ETHICS STATEMENT

The experiments concern visual prediction for manipulation. Plausible generated futures can misrepresent physical outcomes. The reported evaluations do not establish suitability for safety-critical robotic control.

STATEMENT OF AI USE

An AI assistant assisted with literature review, drafting, figure preparation, and experiment analysis. Model measurements originate from evaluation artifacts. The authors remain responsible for the claims, references, and experimental reporting.

REFERENCES

Nicholas M. Boffi, Michael S. Albergo, and Eric Vanden-Eijnden. Flow map matching with stochastic interpolants: A mathematical framework for consistency models. *arXiv preprint arXiv:2406.07507v2*, 2025.

Mingyang Deng, He Li, Tianhong Li, Yilun Du, and Kaiming He. Generative modeling via drifting. *arXiv preprint arXiv:2602.04770v2*, 2026.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *International Conference on Learning Representations*, 2025.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J. Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. *arXiv preprint arXiv:2505.13447*, 2025.

Zhengyang Geng, Yiyang Lu, Zongze Wu, Eli Shechtman, J. Zico Kolter, and Kaiming He. Improved mean flows: On the challenges of fastforward generative models. *arXiv preprint arXiv:2512.02012v2*, 2026.

Yanjiang Guo, Lucy Xiaoyang Shi, Jianyu Chen, and Chelsea Finn. Ctrl-World: A controllable generative world model for robot manipulation. In *International Conference on Learning Representations*, 2026.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. In *Advances in Neural Information Processing Systems*, 2025.

Jin-Young Kim, Hyojun Go, Lea Bogensperger, Julius Erbach, Nikolai Kalischek, Federico Tombari, Konrad Schindler, and Dominik Narnhofer. Understanding, accelerating, and improving meanflow training. In *IEEE and CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B. Tenenbaum. Learning to act from actionless videos through dense correspondences. In *International Conference on Learning Representations*, 2024.

Cheuk-Kit Lee, Florentin Coeurdoux, Yuyuan Chen, Sophia Tang, Peter Potaptchik, Yilun Du, Michael S. Albergo, and Eric Vanden-Eijnden. Beckmann transport models: From autonomous flows to one-step maps. *arXiv preprint arXiv:2608.01692v3*, 2026.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023.

Susie Lu, Haonan Chen, Weirui Ye, and Yilun Du. DriftWorld: Fast world modeling through drifting. *arXiv preprint arXiv:2607.15065v2*, 2026a.

Yiyang Lu, Susie Lu, Qiao Sun, Hanhong Zhao, Zhicheng Jiang, Xianbang Wang, Tianhong Li, Zhengyang Geng, and Kaiming He. One-step latent-free image generation with pixel mean flows. *arXiv preprint arXiv:2601.22158v3*, 2026b.

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

Han Qi, Haocheng Yin, Aris Zhu, Yilun Du, and Heng Yang. Inference-time enhancement of generative robot policies via predictive world modeling. *IEEE Robotics and Automation Letters*, 2026.

Julian Quevedo, Ansh Kumar Sharma, Yixiang Sun, Varad Suryavanshi, Percy Liang, and Sherry Yang. WorldGym: World model as an environment for policy evaluation. *arXiv preprint arXiv:2506.00613*, 2025.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *European Conference on Computer Vision*, 2024.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning*, 2023.

Sophia Tang and Shiyi Wang. Discrete Beckmann transport models for one-step language modeling and reasoning. *arXiv preprint arXiv:2609.15903*, 2026.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

Sherry Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Leslie Kaelbling, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. In *International Conference on Learning Representations*, 2024.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Frédo Durand, and William T. Freeman. Improved distribution matching distillation for fast image synthesis. In *Advances in Neural Information Processing Systems*, 2024.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Fangqi Zhu, Hongtao Wu, Song Guo, Yuxiao Liu, Chilam Cheang, and Tao Kong. IRASim: A fine-grained world model for robot manipulation. In *IEEE and CVF International Conference on Computer Vision*, pp. 9834–9844, 2025.

A Implementation Details

A.1 Tensor Shapes and Action Alignment

History has shape $[B, 3, 4, 96, 96]$. The source and target have shape $[B, 3, F, 96, 96]$, with $F = 4$ for PushT and $F = 2$ for Robomimic Can. Actions have shape $[B, F, A]$, with $A = 2$ and $A = 7$, respectively. The history is flattened into 12 channels and broadcast over future positions. It is concatenated with the three transported RGB channels. Frame-aligned action embeddings condition the residual blocks. The backbone combines spatial and temporal convolutions with spatial attention. It is not a temporal-attention transformer.

For a PushT window using observed images at indices 0 through 3, future images are at indices 4 through 7 and controls are at indices 3 through 6. Absolute planar actions are normalized by $\frac{2a}{511} - 1$. Images are normalized to $[-1, 1]$. A three-frame planning tail is evaluated at its actual length. Padding future actions and then discarding frames could change valid outputs because the inherited temporal mixing is not strictly causal.

The residual map is $T_\theta(x, c) = x + U_\theta(x, c)$. The input tensor differentiated by the JVP is x only. Both h and a remain fixed in its closure. The implementation preserves the forward-mode derivative even though the resulting target is stopped for reverse-mode differentiation. Paths incompatible with forward-mode differentiation are bypassed rather than replaced with an approximate finite difference. This distinction matters for execution cost.

A.2 Gradient Balancing and Optimization

Let w denote parameters of the output spatial projection. Define

$$g_{\mathrm{tr}} = \nabla_w \mathcal{L}_{\mathrm{tr}}, \qquad g_{\mathrm{bd}} = \nabla_w (\mathcal{L}_{\mathrm{bd}} + \mathcal{L}_{\mathrm{near}}). \tag{9}$$

We average each gradient vector across distributed ranks before taking its Euclidean norm. When both norms exceed 10^{-8}, the detached balance is

$$\lambda = \mathrm{sg}\left(\mathrm{clip}\left(\frac{\|g_{\mathrm{bd}}\|_2}{\|g_{\mathrm{tr}}\|_2}, 10^{-3}, 10^{3} \right) \right). \tag{10}$$

When either finite norm is at most 10^{-8}, the coefficient is one. Nonfinite gradients cause a failure rather than a silent substitution. The fallback prevents an initially zero boundary gradient from disabling transport. Source and endpoint auxiliary terms are not inserted into this original balancing group.

The stabilized map uses Muon for hidden weights and AdamW for output and auxiliary parameters. The original target learning rates are 10^{-3} and 10^{-4}. The completed motion-aware continuation uses one quarter of these rates. Warmup lasts 500 updates from one percent of the target rate. The global gradient norm is clipped at two, and EMA decay is 0.999. Training and the recorded full-update comparison use full precision. Each continuation explicitly initializes from its parent's EMA and resets the continuation's optimizer and scheduler. Resuming the same continuation instead restores its own model, optimizer, scheduler, EMA, RNG states, sampler position, and update clock.

The matched DriftWorld backbone does not imply identical optimization. Its recorded runs use AdamW, smaller window batches, and 16 generated negatives per PushT condition or 32 per Robomimic Can condition. Kernel temperatures are 0.02, 0.05, and 0.2. These differences are part of the complete implementation comparison and prevent attributing every speed or quality difference to a single operation.

A.3 Exact Endpoint Regularizers

For example i, let $y_{i,0}$ be the last recorded image before the target chunk. The target-defined moving mask is

$$m_{i,f,u,v} = \mathbf{1}\left[\max_{r \in \{1,2,3\}} |y_{i,r,f,u,v} - y_{i,r,0,u,v}| > 0.05 \right]. \tag{11}$$

The RGB channel index is r, future position is f, and spatial coordinates are (u, v). Broadcast the mask over channels and let mean average the indicated tensor coordinates. The training loss is

$$\mathcal{L}_{\mathrm{move}} = \frac{1}{B} \sum_i \frac{\mathrm{mean}\left[m_i (p_i - y_i)^2\right]}{\max(\mathrm{mean}(m_i), 0.05)}. \tag{12}$$

The floor limits the additional per-coordinate weight induced by the coefficient 0.1 to at most two. The other terms are

$$\mathcal{L}_{\mathrm{temp}} = \mathrm{mean}_{i,r,f=2:F,u,v} \left[\left((p_{i,r,f,u,v} - p_{i,r,f-1,u,v}) - (y_{i,r,f,u,v} - y_{i,r,f-1,u,v}) \right)^2 \right], \tag{13}$$

$$\mathcal{L}_{\mathrm{range}} = \mathrm{mean}\left[\max(|p| - 1, 0)^2\right]. \tag{14}$$

The temporal term concerns within-chunk differences. It does not force a repaired endpoint to preserve the offset of an erroneous generated preceding frame.

The recovery branch uses a separate valid 16-frame training sequence. Robomimic Can generates between one and seven two-frame prefix chunks before a two-frame endpoint. PushT generates between one and three four-frame prefix chunks and alternates four-frame and three-frame endpoints. The depth curriculum expands over 1,000 completed updates. Prefix generation is detached, raw, and conditioned on aligned recorded actions. No blend with the true history or inference clamp is introduced. The branch is active every fourth update, with weight 0.25 on active updates. It is not multiplied by four to compensate for its frequency. All losses use one final backward pass and one optimizer update.

The clean-window dataset retains short episodes that cannot support a 16-frame sequence. The recovery sequence is obtained from a separate training-only valid-sequence index. No sample crosses an episode boundary. This regularizer does not create labeled counterfactual simulator transitions.

A.4 Source-Loss Variance Decomposition

Fix c and assume finite second moments and conditional independence of z and y given c. Write $P = T_\theta(z, c)$, $P = \mu_P + \varepsilon_P$, and $y = \mu_y + \varepsilon_y$. Expanding the squared difference gives

$$\mathbb{E}\|P - y\|_D^2 = \|\mu_P - \mu_y\|_D^2 + \mathbb{E}\|\varepsilon_P\|_D^2 + \mathbb{E}\|\varepsilon_y\|_D^2 - \frac{2}{D}\mathbb{E}\langle \varepsilon_P, \varepsilon_y \rangle \tag{15}$$

$$= \|\mu_P - \mu_y\|_D^2 + \mathrm{Var}_D(P \mid c) + \mathrm{Var}_D(y \mid c). \tag{16}$$

The final cross term is zero by conditional independence. This is a property of independent paired squared error, not a new transport theorem. For multimodal futures, reducing source-induced variation can improve paired MSE while degrading distributional coverage. Conversely, source-induced variation that does not correspond to possible outcomes is not useful uncertainty. Both accuracy and distributional scoring are therefore needed.

A.5 Training Schedule

Algorithm 1 makes the distinction between a stopped transport target and trainable endpoint predictions explicit. The base map first learns $\mathcal{L}_{\mathrm{BTM}}$. Source supervision is then introduced with its 500-update ramp. The completed motion-aware continuation starts from that source-anchored EMA with source weight already one. It ramps only the added endpoint and recovery terms.

B Additional Results and Training Lineage

B.1 Completed Training Recipes

The complete model denotes motion-aware endpoint supervision with detached generated-history recovery. The source anchor is its parent. Learned direction is a separate continuation from that parent. It is not part of the complete recipe. Robomimic Can uses 5,000 updates from random initialization, 5,000 further source-anchoring updates, and 5,000 motion-aware updates, for 15,000 total. The model sees eight primary conditioning windows per update. DriftWorld uses 20,000

updates with two conditioning windows per update. Its generated population has size 32 for each condition.

PushT source anchoring inherits a substantially longer development history. Table 2 reports approximately 141,700 cumulative updates for the stabilized map, 163,300 for source anchoring, and 183,000 for learned direction. DriftWorld has 39,350 updates. Consequently, the completed PushT prediction results do not demonstrate a lower training budget. The initial BTM port was evaluated on PushT only.

The connected-recovery alternative on Robomimic Can is not an isolated detachment ablation. It uses two connected predicted chunks, one recovery example per GPU, the original learning rates, a 500-update ramp, a stronger recovery moving-region coefficient, and no temporal or range losses. The main recipe uses detached prefixes with varying depth, a four-example recovery batch per GPU, quarter learning rates, a 1,000-update ramp, and all endpoint losses. Both continue the same source-anchored parent. Their comparison measures two complete training recipes.

Table 3: Additional final Robomimic Can measurements. All methods use the same 30 validation episodes. Best values are in bold.

	64 total frames		Full episode		
Method	LPIPS	Moving MSE	LPIPS	Moving MSE	Raw MSE
DriftWorld	.0682	.0769	.1463	.1316	.0256
Stabilized map	.0649	.1452	.1914	.2669	.1159
Source-anchored	.0225	.0576	.1390	.1710	.0940
Motion-aware (ours)	**.0090**	**.0316**	**.0334**	**.0659**	**.0112**
Learned direction	.0854	.1417	.2223	.2256	.0759
Connected recovery	.0215	.0488	.0647	.0801	.0146

B.2 Paired Episode Comparisons

The saved final evaluation records include individual metrics for the same 30 Robomimic Can episodes under each horizon. We checked the cohort, EMA selection, batch size, seed, preprocessing, and aggregation before matching episode IDs. Table 4 uses 20,000 paired bootstrap resamples of these episodes. It gives uncertainty across validation episodes for fixed trained models. It is not a confidence interval across training repetitions.

Table 4: Paired differences from DriftWorld on Robomimic Can. Negative values favor the listed map. Wins count episodes with strictly lower error out of 30. Intervals are 95% episode-bootstrap intervals for fixed checkpoints.

Method	Horizon	Metric	Difference	95% interval	Wins
Source-anchored	64 frames	LPIPS	-0.0456	$[-0.0481, -0.0432]$	30
Source-anchored	64 frames	Moving MSE	-0.0193	$[-0.0293, -0.0090]$	21
Source-anchored	64 frames	MSE	-0.0035	$[-0.0046, -0.0025]$	25
Source-anchored	Full episode	MSE	$+0.0684$	$[+0.0200, +0.1309]$	12
Source-anchored	Full episode	LPIPS	-0.0073	$[-0.0306, +0.0198]$	21
Motion-aware (ours)	64 frames	LPIPS	-0.0592	$[-0.0615, -0.0568]$	30
Motion-aware (ours)	64 frames	Moving MSE	-0.0453	$[-0.0529, -0.0378]$	29
Motion-aware (ours)	64 frames	MSE	-0.0073	$[-0.0082, -0.0064]$	30
Motion-aware (ours)	Full episode	MSE	-0.0144	$[-0.0160, -0.0128]$	30
Motion-aware (ours)	Full episode	LPIPS	-0.1129	$[-0.1218, -0.1043]$	30

B.3 Cumulative Learning Cost

For each training run, successive 20-update intervals cover the entire recorded update sequence without gaps or duplicate updates. The map lineage has 250 intervals for each continuation, while DriftWorld has 1,000. The corresponding clocks are accumulated before plotting evaluation checkpoints. They include training-loop work but exclude inter-evaluation waits, interruptions, and data

Table 5: Robomimic Can cumulative training accounting. Hours are summed from complete logged training-loop intervals. Each run uses two training GPUs. All ancestors are counted once.

Recipe	Added updates	Cumulative updates	Added hours	Total hours
Stabilized map	5,000	5,000	1.141	1.141
Source-anchored	5,000	10,000	1.497	2.639
Motion-aware (ours)	5,000	15,000	1.683	4.322
DriftWorld	20,000	20,000	10.194	10.194

preparation. Doubling these hours gives device-hours assigned to the two-GPU training loops, not total reserved cluster time. Evaluator time and development experiments are not included in this cost-to-train comparison.

The latest evaluated DriftWorld checkpoint inside five hours is at 8,000 updates and 4.088 hours. The completed motion-aware model is at 4.322 hours. Their 64-frame LPIPS values are 0.12574 and 0.00898. The next DriftWorld evaluation occurs at 6.122 hours. Figure 1 therefore shows observed checkpoints rather than an interpolated equal-time score.

Complete-update profiling. Table 6 uses fresh disposable models with random initialization and the saved training configurations, overriding only the primary batch to one window per GPU. The training clock advances through the mature schedule after its ramps. Eight warmup updates precede 56 measured updates. The 14 active recovery updates visit all seven Robomimic Can prefix depths twice. Primary windows are the throughput denominator, while generation and losses for recovery sequences are included in the numerator's elapsed time.

Each update includes device transfer, all model passes, the JVP, gradient balancing, auxiliary losses, backward propagation, distributed reduction, clipping, optimizer, scheduler, and rank-zero EMA. Data-loader waiting is outside the timed region. CUDA synchronization bounds each update, and the slower rank determines its duration. Peak memory is the maximum allocated PyTorch memory across ranks after warmup, including live model and optimizer state. It is not total device reservation. Measurements use PyTorch 2.11.0, CUDA 12.8, full precision, and disabled TF32 on two A800 80GB GPUs. Profile models are discarded and are not used for accuracy measurements.

B.4 Learned Directions as a Diagnostic Alternative

A learned direction replaces a samplewise chord in the map target by an estimate $b_\psi(x_s, c)$ of its conditional mean. The auxiliary squared regression target is $d = y - z$. The map target uses a detached blend

$$\widetilde{d} = (1 - \alpha_k)d + \alpha_k \,\mathrm{sg}(b_\psi(x_s, c)), \qquad u = \mathrm{sg}\left(T_\theta(x_s, c) + 0.25 J_x T_\theta(x_s, c)\widetilde{d}\right). \tag{17}$$

The direction loss updates shared features and a separate output projection. Its squared-error regression has weight one and is active immediately. The tangent blend is $\alpha_k = \min(1, \max(0, (k - 500)/500))$, where k counts completed continuation updates. The direction head is unused during sampling. The diagram in Figure 7 distinguishes individual sampled chords from their learned conditional mean. It is not a visualization of measured video-space vectors. The completed learned-direction continuation helps PushT visual prediction but degrades Robomimic Can metrics (Table 2). It does not improve final PushT action-selection IoU, but the completed policy-evaluation study reaches Pearson correlation 0.9743 across seven policies. These distinct outcomes support reporting it as a task-dependent alternative rather than a required component of the final recipe.

B.5 Repeated Map Application

A retained fixed-window Robomimic Can source-map probe applies the map to the same future with unchanged history and actions. One, two, and four applications give LPIPS 0.002673, 0.002508, and 0.004643, with moving-region MSE 0.012135, 0.012490, and 0.015419. These are separate fixed-window measurements, not the 64-frame benchmark. Additional calls do not advance physical time. The deterioration at four calls illustrates why terminal-map geometry alone does not establish useful refinement for a finite trained network.

Table 6: **Complete-update cost at a matched batch size.** Robomimic Can, two A800 80GB GPUs, one conditioning window per GPU, two predicted frames, full precision. Eight warmup updates precede 56 measured updates. The mature motion-aware schedule includes 14 recovery updates spanning all seven prefix depths twice. Throughput counts primary conditioning windows. Every auxiliary computation is included in time and memory.

Method	Mean seconds	P95 seconds	Windows per second	Peak GiB
DriftWorld	1.841	1.843	1.09	36.07
Source-anchored	0.496	0.509	4.03	6.12
Ours, complete recipe	0.553	0.739	3.62	7.77

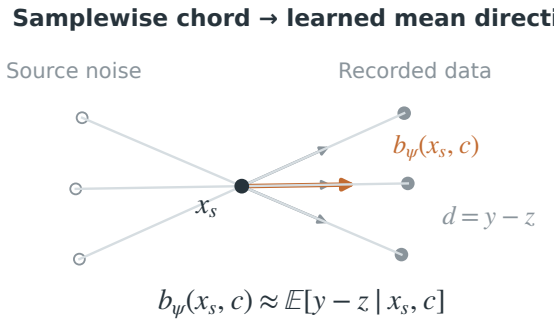

Figure 7: **Changing the training tangent.** Several noise–data chords can pass through one interpolated input. Gray arrows show their sampled directions. The orange arrow represents a learned conditional-mean estimate. This changes the JVP used to form a training target. It does not add an inference step.

B.6 Interim PushT Motion-Aware Results

At the evidence cutoff of September 25, 2026, at 09:27 UTC, the motion-aware PushT continuation has completed three of five planned epochs, or 11,805 added updates. Its 64-frame LPIPS is 0.0644, moving-region MSE 0.2786, raw MSE 0.0237, and full-episode MSE 0.0359. The latest connected-recovery alternative has completed four of five epochs, or 15,740 added updates. Its LPIPS is 0.0846, raw MSE 0.0228, and full-episode MSE 0.0346. These checkpoints have different added training durations and recipes. Neither is substituted for a completed final model in the main table. The first motion-aware epoch gives native GPC-RANK IoU 0.688, with a paired difference interval from DriftWorld of $[-0.074, 0.031]$. These intermediate results do not establish a planning improvement.

B.7 Endpoint-Map Diagnostic

The diagnostic in Figure 4 uses the first, middle, and last valid two-frame prediction window in each of the 30 Robomimic Can validation episodes. Each window has four real history frames. Four Gaussian source tensors with standard deviation 1.35 are shared across checkpoints and interpolation coefficients. The coefficients are 0, 0.25, 0.5, 0.75, and 1. Every query is an independent map call, with history and actions fixed. Neither the interpolated inputs nor their successive outputs form an inferred autonomous trajectory.

The table reports averages over sources within windows, windows within episodes, and episodes. Source variance is the mean-coordinate variance of four outputs using denominator four. The finite-source identity decomposes empirical MSE exactly into squared error of the mean prediction and this variance. Clean-boundary variance is zero because $x_1 = y$ is identical for all source draws. All errors use raw normalized RGB. The moving mask is derived from each recorded target and the last real history image, with threshold 0.05 in any channel. This diagnostic uses no generated history and no weight updates.

Table 7: Conditional endpoint behavior on 90 fixed Robomimic Can windows. Source and boundary errors measure different inputs to the same map. Lower is better for the reconstruction columns. Lower source variance alone does not establish better uncertainty modeling.

Method	Source MSE	Source moving MSE	Source variance	Boundary MSE
Stabilized BTM	0.002831	0.029575	0.001458	0.000105
Source-anchored	0.000823	0.013851	0.000192	0.000096
Full recipe	0.000503	0.008832	0.000034	0.000025
Learned direction	0.001170	0.012249	0.000440	0.000112

C Evaluation Protocols

C.1 Prediction and Metric Preprocessing

One native chunk contains two Robomimic Can frames or four PushT frames. The 64-total-frame benchmark starts from four real history frames and scores only the next 60 generated frames. Subsequent histories use raw predictions without clipping. Full-episode evaluation continues through the remaining valid recorded episode. Exact-length batches avoid padding future action sequences across episode endpoints. A partial final chunk is evaluated at its actual length, because the inherited temporal mixing is not strictly causal.

Raw MSE is computed in normalized RGB coordinates without clipping. LPIPS uses the retained AlexNet evaluator with inputs clipped to normalized image bounds. PSNR and SSIM use clipped and quantized RGB images. SSIM uses the channel-aware implementation. The moving-region metric selects ground-truth pixels differing from the final real initial observation by more than 0.05 in any normalized RGB channel. An episode's moving error averages over the selected coordinates, and empty supports are excluded. The final Robomimic Can records contain no empty moving-region episodes. This evaluation metric is distinct from the bounded training-mask loss in Appendix A.

Pixel and perceptual metrics average generated frames within each episode. Moving-region error pools the selected RGB coordinates within each episode. All metrics then weight episodes equally. The final Robomimic Can comparisons use EMA weights, seed 314, batch size 16, and the same exact-length grouping and episode order. The 64-frame cohort has 1,800 generated frames and the full-episode cohort 6,450, each from 30 episodes. These shared records support the paired comparisons in Table 4.

C.2 One-Call Real-History Probe

The additional evaluation in Table 8 fixes three valid starts in each Robomimic Can validation episode before inference. For an episode of length L, the zero-based starts are 0, $\lfloor (L-6)/2 \rfloor$, and $L-6$. Four recorded history frames condition exactly one call predicting the next two frames. Actions align with the transition from the last history frame. The sampler receives seed $250926+3i+j$, with zero-based episode position i and window position j, reset for every method. The checkpoint's source scale is retained. Raw predictions are scored before any feedback or display clipping.

Each metric is averaged over the three windows and then over the 30 episodes. The moving mask is computed separately for each window relative to its last observed frame, using the same 0.05 threshold. Copying that observation into both future positions is a nonlearned baseline. Against DriftWorld, the complete model's paired episode-bootstrap differences are -0.000970 MSE with interval $[-0.001034, -0.000911]$, -0.00636 moving-region MSE with interval $[-0.00688, -0.00585]$, and -0.00634 LPIPS with interval $[-0.00667, -0.00602]$. The bootstrap uses 20,000 episode resamples and seed 250926. All three errors are lower on every episode. These real-history measurements are reported separately from autoregressive results.

C.3 Statistical Scope

The reported training runs are single trajectories. Successive checkpoints estimate learning progress, not training-seed uncertainty. For Table 4, episode IDs are paired across models and resampled with

Table 8: **One-call endpoint prediction on Robomimic Can.** Three windows per validation episode are fixed at its first, middle, and last valid starts. Each model predicts two frames from four real history frames. Scores average three windows within each of 30 episodes. Lower is better.

Predictor	Raw MSE	Moving MSE	LPIPS
Copy last observation	.002407	.04586	.00434
DriftWorld	.001473	.01522	.00923
Source-anchored	.000811	.01374	.00437
Motion-aware (ours)	**.000504**	**.00886**	**.00289**

replacement 20,000 times using bootstrap seed 250925. The interval is the 2.5th to 97.5th percentile of mean paired differences. Resampling episodes preserves dependence among frames in a rollout. It does not remove bias from repeated use of the validation cohort during development.

The fresh qualitative probe instead uses batch size one and resets its generator to $250925 + i$ for validation position i. Models receive identical standardized noise draws, scaled according to their saved source distributions. This produces controlled visual comparisons but need not reproduce the batched seed-314 predictions. Its per-frame traces are reported separately. No aggregate metric in Table 1 is replaced by the mean over the three illustrated episodes.

C.4 Action Selection

Native PushT GPC-RANK uses the released frozen diffusion policy at epoch 100 and image-based outcome readers. It proposes 50 candidate action sequences, predicts 15 steps through four calls of lengths $4, 4, 4, 3$, and scores the terminal predicted image. The controller executes eight actions from the selected proposal. The evaluation cohort contains 50 fixed environment seeds. They are evaluation cases, not training replications.

The final learned-direction model's mean difference from DriftWorld is -0.045 IoU, with paired interval $[-0.094, 0.001]$. This is a lower point estimate without an established difference at this interval level. A reader's score on a generated image, its score on a real rendered image, and the simulator's physical task score are separate quantities. The benchmark retains the released readers for comparability. Robomimic Can has no corresponding controller evaluation in this setup.

C.5 Policy Evaluation

We use the seven released diffusion policies at training epochs 50, 100, 150, 200, 250, 300, and 650. Each policy is rolled out in the model and simulator for 300 trials with base seed 100000 and execution horizon eight. Each point in Figure 6b is a policy-level mean. Table 9 gives the underlying means, and Table 10 separates ordering from absolute score error. The archived simulator mean

Table 9: PushT policy evaluation over 300 trials per policy. The simulator column uses the source-anchored reference. The starred entry differs for local DriftWorld as detailed in the text. Model columns abbreviate source anchoring and learned direction.

Policy epoch	Simulator	Local DriftWorld	Source	Direction	Released DriftWorld
50	0.6150	0.5642	0.5327	0.5681	0.5884
100	0.6034	0.5676	0.5462	0.5633	0.5762
150	0.6049	0.5548	0.5297	0.5561	0.5715
200	0.5701	0.5557	0.5176	0.5400	0.5565
250	0.5486	0.5383	0.5112	0.5298	0.5233
300	0.5684	0.5347	0.5063	0.5336	0.5524
650	0.5498*	0.5337	0.5048	0.5207	0.5496

for the epoch-650 policy is 0.55308 in the locally trained DriftWorld evaluation and 0.54982 in the source-anchored, learned-direction, and released-checkpoint evaluations. The other six simulator means agree. Each correlation and error uses its own archived simulator reference. We therefore report descriptive comparisons and do not claim a fully paired significance test between the models.

Table 10: Ordering and absolute accuracy across seven policies. Each correlation uses its own archived simulator reference. All models underpredict the simulator means.

World model	Pearson $r \uparrow$	Spearman $\rho \uparrow$	Mean absolute gap $\downarrow$
Local DriftWorld	.8565	.7143	.0306
Source-anchored	.8708	.7857	.0588
Learned direction	.9743	.9286	.0355
Released DriftWorld	.9255	.9643	.0203

The released DriftWorld checkpoint is retained as a control with different training provenance. The completed learned-direction study uses the same seven policies and matches the source-anchored simulator references. Its higher Pearson correlation coexists with lower Spearman correlation and a larger absolute gap than the released control. The completed motion-aware Robomimic Can model is not evaluated by these PushT policies.

D MATCHED QUALITATIVE COMPARISONS

The following full rollouts use final Robomimic Can checkpoints at validation positions one, sixteen, and thirty, fixed before inference. Each comparison shares four real initial frames, recorded actions, and standardized noise draws. Physical frame indices match across methods. The last column is the episode's final frame. Exported RGB is clipped, while feedback and error traces use raw predictions. Images are unretouched and traces are unsmoothed. These examples complement the complete validation cohort and do not establish physical correctness or calibrated uncertainty.

Figure 8: First Robomimic Can validation episode, 142 total frames. The source-anchored model develops a distorted scene late in the rollout. The complete recipe retains more of the recorded configuration in this example, while its final prediction still differs from ground truth.

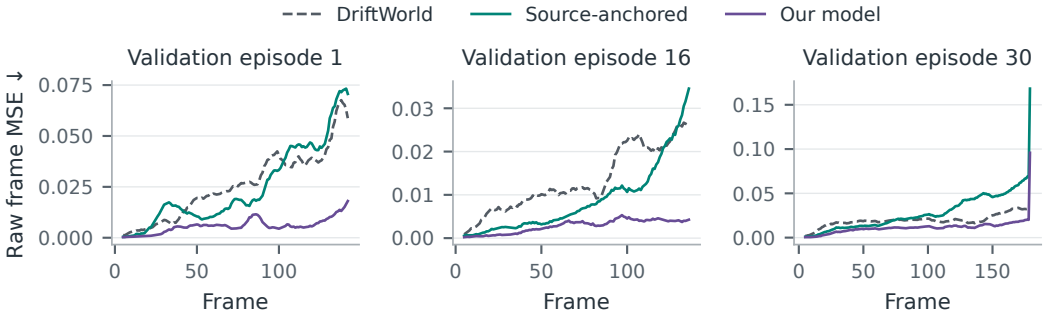

Figure 9: Raw per-frame MSE for every generated frame in the three illustrated episodes. Each line is an actual rollout, without smoothing. These traces belong to the fresh batch-one visualization protocol, not the batched seed-314 cohort used for the main table.

Figure 10: Middle Robomimic Can validation episode, 136 total frames. The same checkpoint identities and display conventions are used. The example is retained independently of its measured model ordering.

Figure 11: Last Robomimic Can validation episode, 179 total frames. All rows use recorded actions throughout. Even low mean error can coexist with a mismatch in the final object or gripper configuration.

D.1 PushT Appearance and Pose

The following retained comparison uses actual model outputs from the earlier completed PushT evaluation. It compares DriftWorld with the learned-direction map, not the pending motion-aware recipe. Recognizable object shape and correct object pose are different properties. The example illustrates that distinction without attributing the action-selection result to one visual failure mode.

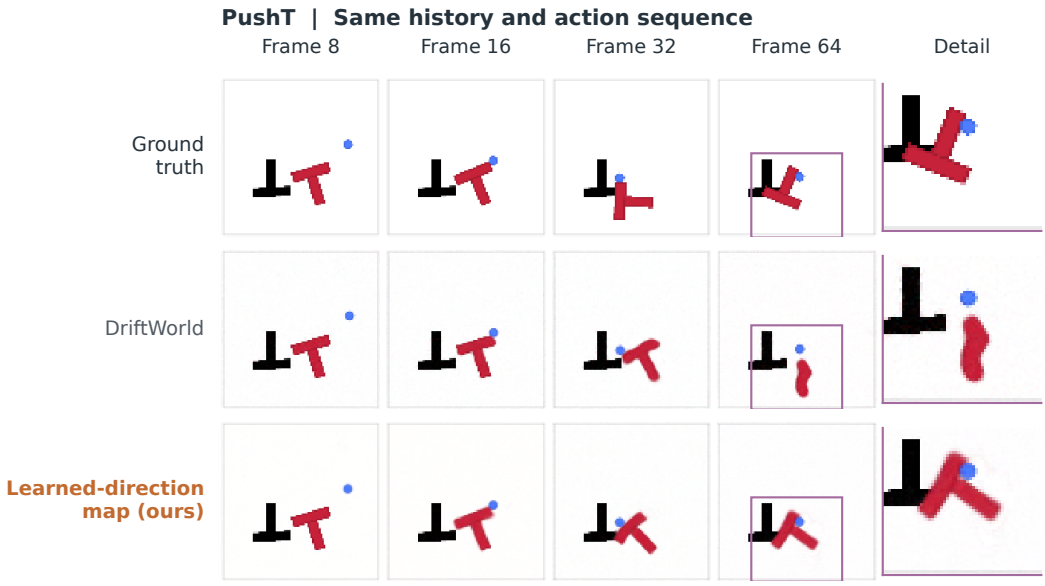

Figure 12: Matched PushT trajectory with recorded future, DriftWorld, and the learned-direction map. A recognizable T-shaped object can still have incorrect position or orientation. The displayed comparison is retained as qualitative context for Figure 6a.

E Additional Baselines and Protocol Compatibility

E.1 Released GPC Evaluation

The official GPC code and final released PushT world-model checkpoint are evaluated on exactly the 388 validation episodes used by our 64-frame benchmark (Qi et al., 2026). The checkpoint is used without fine-tuning.[1] Its denoiser has approximately 8.7 million parameters and predicts one frame from four preceding images and four aligned controls. To predict frame t, it receives images $t-4$ through $t-1$ and controls $t-4$ through $t-1$. The first three of these controls precede the newly proposed control. Beckmann World Models instead condition each native chunk on four images and its four future controls. Both receive recorded controls with the same absolute-coordinate normalization.

The released sampler uses three first-order denoising evaluations per generated frame, with noise levels determined by its published implementation. The official loader supplies images in $[0, 1]$. GPC receives this native range, and its numerical predictions are fed back unchanged. Predictions are converted by $2x - 1$ only at the common metric interface. Its native per-denoising-step RGB clipping and quantization are preserved. No new clamp is added to any model. The GPC sampler starts from its own standard Gaussian source. Our source scale remains 1.35. Consequently, a 60-frame rollout requires 180 GPC denoiser evaluations and 15 Beckmann map evaluations. Both count actual network invocations, not a nominal sampler step for an entire video.

The comparison uses seed 314, batch size 16, four real initial images, 60 generated frames, and the existing metric implementation. LPIPS uses AlexNet with clipped generated RGB. MSE uses the numerical predictions before display processing. SSIM and PSNR use the established quantized image convention. Moving-region error uses the same target-defined mask as the primary evaluator. Metrics average within episodes and then equally across episodes. Public GPC training data include the source corpus from which our split was constructed. Its release does not certify exclusion of these 388 episodes. The released row therefore measures prediction on a common evaluation cohort, not equal-data generalization or matched training cost.

GPC reaches mean raw MSE 0.01920, LPIPS 0.03238, moving-region MSE 0.24456, SSIM 0.96579, and PSNR 27.002 dB. These are newly computed values, distinct from the published reference table.

Inference timing. All timing inputs are already on the GPU. One query predicts four future frames at batch size one or sixteen. GPC makes four autoregressive sampler calls, each with three denoising evaluations. The one-step models make one native four-frame call. After eight warmup queries, 20 complete queries are timed with CUDA synchronization. The profile uses full precision with TF32 disabled on one A800 80GB GPU. It excludes image metrics, data loading, and display export. The evaluator's cooperative lock prevents overlap with its other evaluation jobs. This profile is distinct from the two-GPU training-update measurements.

E.2 Published Benchmark Results

DriftWorld reports GPC diffusion, AVDC, Ctrl-World, and an MSE-trained U-Net on PushT, together with GPC and Ctrl-World on single-view Robomimic Can (Lu et al., 2026a, Qi et al., 2026, Ko et al., 2024, Guo et al., 2026). AVDC and the MSE U-Net have no corresponding Robomimic Can row in that comparison. IRASim, VDM, LVDM, and WorldGym are evaluated on other datasets and are not assigned inferred PushT or Robomimic Can values.

Evaluation populations. Published PushT scores use 1,000 examples. Our held-out comparison uses 388 eligible 64-frame episodes and 400 full episodes. The released DriftWorld entrypoint obtains windows or full episodes from its configured training-data directory, without our episode-level split. Its fixed-length loader samples windows, whereas ours begins each eligible episode at its first four frames. Published Robomimic Can scores concern full-video evaluation with unspecified correspondence to our 30 validation episodes. These reported values are therefore retained as sepa-

[1] Official GPC code, revision a96146e, and released checkpoints, revision c16a467. We use the final denoiser from the second world-model training phase.

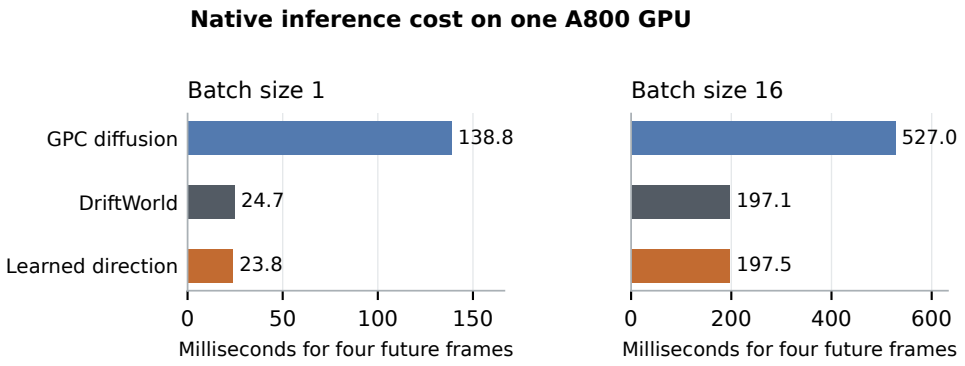

Figure 13: **Native inference cost at a common output length.** Each query predicts four PushT frames. Released GPC uses 12 denoiser evaluations. DriftWorld and the learned-direction model each use one network evaluation. Bars average 20 synchronized queries after eight warmups on one A800 GPU. Batch size counts independent queries.

Table 11: **Published benchmark context from DriftWorld.** Values are transcribed from Tables 1–2 of Lu et al. (2026a). They use a different evaluation population from our split. All images are 96×96. Dashes denote unreported MSE, not missing model evaluations.

Task and horizon	Method	MSE ↓	SSIM ↑	PSNR ↑	LPIPS ↓
PushT, 64 frames	GPC diffusion	.0033	.9717	33.1086	.0239
	AVDC	.0045	.9862	30.4990	.0100
	Ctrl-World	.0540	.8914	18.0705	.1844
	MSE U-Net	.0105	.9704	28.0578	.0225
	DriftWorld	.0020	.9941	34.7751	.0035
PushT, full episode	GPC diffusion	.0033	.9713	32.3692	.0278
	AVDC	.0063	.9825	29.8518	.0143
	Ctrl-World	.0576	.8854	18.0638	.2513
	MSE U-Net	.0150	.9632	26.5779	.0389
	DriftWorld	.0061	.9869	32.6608	.0124
Robomimic Can, full episode	GPC diffusion	—	.8917	22.9822	.0711
	Ctrl-World	—	.8801	22.6784	.0817
	DriftWorld	—	.8858	28.8460	.0614

rate reference measurements. Matching metric names and image size alone does not justify a joint ranking with our main results.

E.3 Matched Diffusion and One-Step Rollouts

Figures 14–16 compare the released diffusion model with our task-matched DriftWorld baseline and the completed learned-direction model. The examples are the first, middle, and last eligible validation episodes, fixed before inspecting model quality. Every row receives identical recorded images and controls. Each model uses its native sampler with a fixed seed for the example. Common display clipping follows metric evaluation. No generated image is retouched.

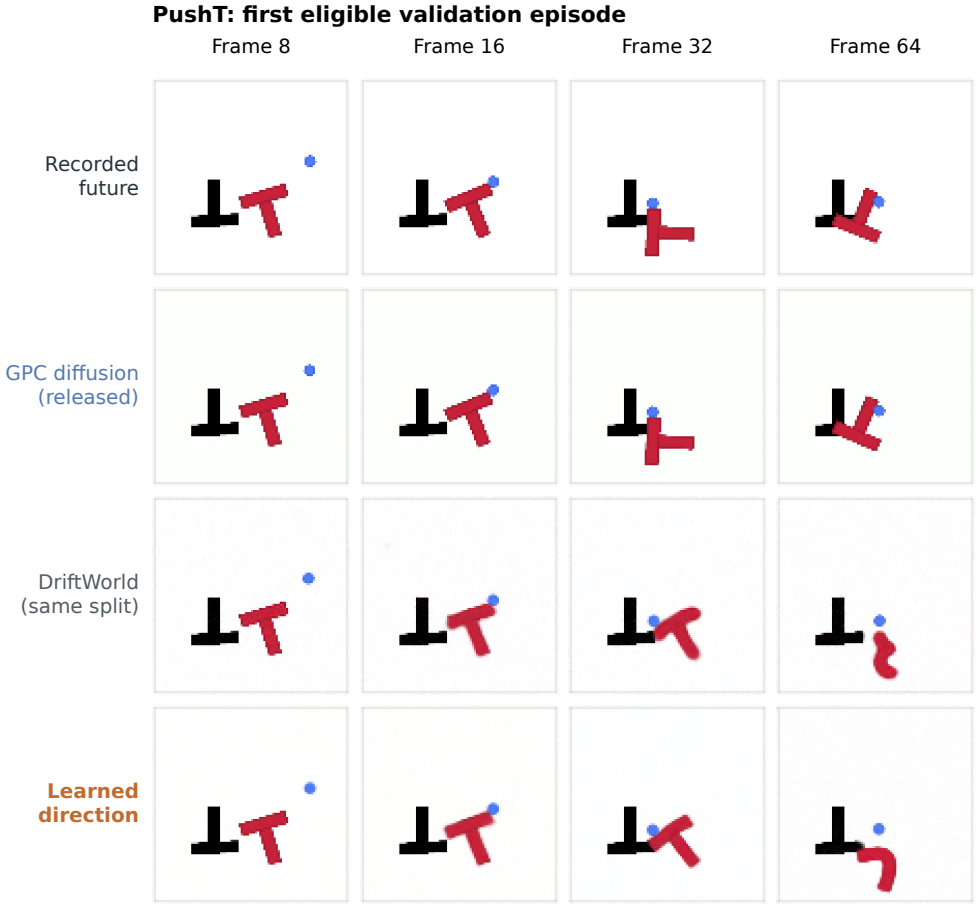

Figure 14: **PushT comparison on the first eligible validation episode.** The released GPC diffusion model uses three denoising evaluations per frame. DriftWorld and the learned-direction Beckmann model use one evaluation per four-frame chunk. The same first four recorded images initialize each 60-frame rollout. Columns show total-sequence frame numbers. Training provenance differs for the released model.

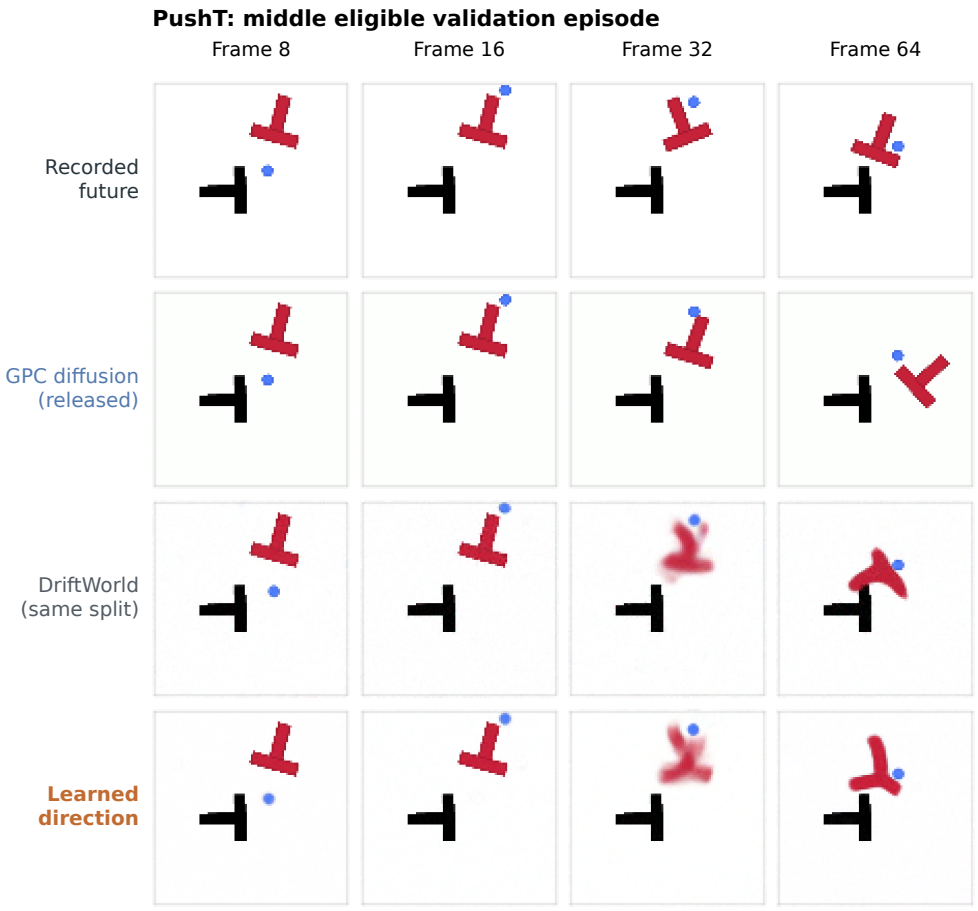

Figure 15: **PushT comparison on the middle eligible validation episode.** The released GPC diffusion model uses three denoising evaluations per frame. DriftWorld and the learned-direction Beckmann model use one evaluation per four-frame chunk. The same first four recorded images initialize each 60-frame rollout. Columns show total-sequence frame numbers. Training provenance differs for the released model.

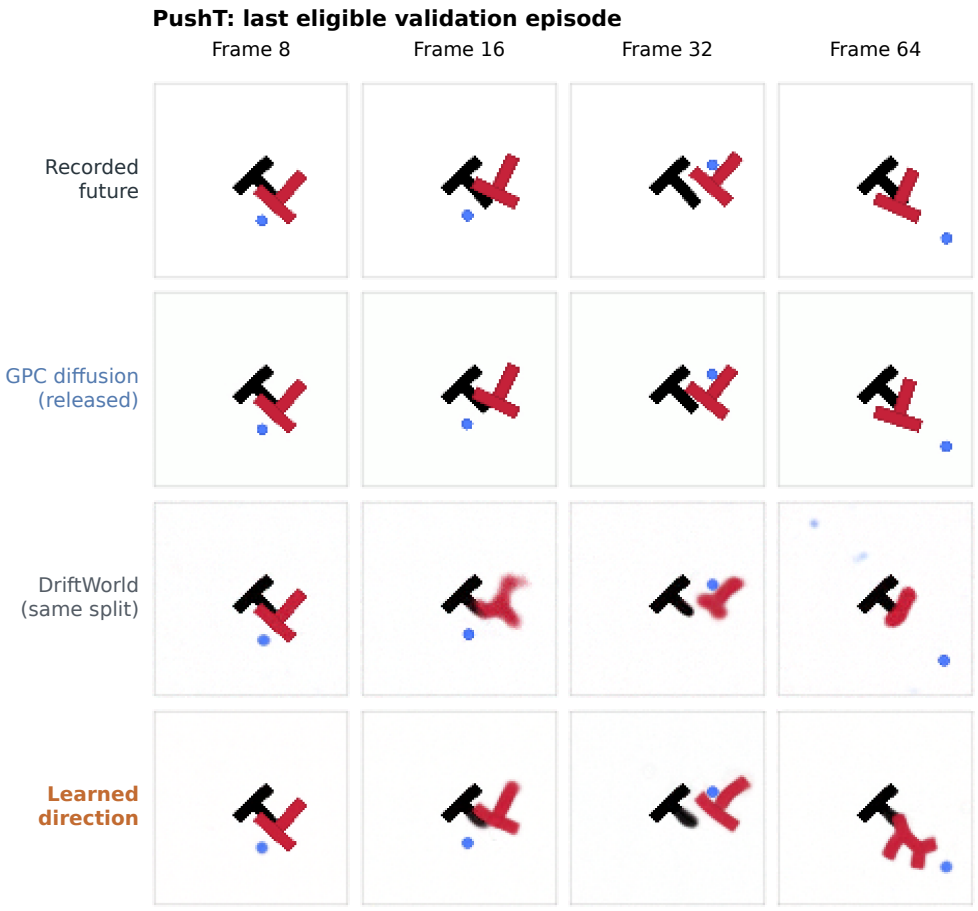

Figure 16: **PushT comparison on the last eligible validation episode.** The released GPC diffusion model uses three denoising evaluations per frame. DriftWorld and the learned-direction Beckmann model use one evaluation per four-frame chunk. The same first four recorded images initialize each 60-frame rollout. Columns show total-sequence frame numbers. Training provenance differs for the released model.