Dynamics-Aligned Flow Matching Policy for Robot Learning

Dohyeok Lee[1], Jung Min Lee[1], Munkyung Kim[1], Seokhun Ju[1],
Seungyub Han[1], Jin Woo Koo[1], Jungwoo Lee[1*]

Abstract

Behavior cloning methods for robot learning suffer from poor generalization due to limited data support beyond expert demonstrations. While recent approaches leverage video prediction models to implicitly capture dynamics, they lack explicit action conditioning, leading to averaged predictions over actions that lose critical dynamics information. We propose a Dynamics-Aligned Flow Matching Policy that integrates dynamics predictions into policy learning through iterative flow generation. Our method introduces a novel architecture where policy and dynamics models share intermediate generation samples during training, enabling self-correction and improved generalization. Theoretical analysis demonstrates that conditioning on predicted dynamics leads to improved approximation to optimal actions, with empirical validation on Robomimic benchmarks.

1. Introduction

Behavior cloning (BC) has proven effective for learning robotic manipulation policies using diffusion models [8], yet it faces fundamental limitations in generalization beyond training scenarios. The core challenge lies in the distribution mismatch between expert demonstrations and real-world deployment environments. While diffusion-based policies [9, 26] have shown promising results, they remain susceptible to overfitting when trained solely on expert data due to limited data support outside the optimal trajectory distribution.

Recent works [3, 7, 10, 13, 14, 24, 25] attempt to address this by incorporating video prediction models to implicitly capture environmental dynamics. However, these approaches predict future observations without action conditioning, essentially modeling $p(o_{t+1}|o_t) = \int p(o_{t+1}|o_t, a_t)\pi^*(a_t|o_t)da_t$. This averaging over different action outcomes loses critical dynamics information. Moreover, video prediction models trained exclusively on expert demonstrations tend to overfit to the limited trajectory distribution, leading to poor performance on out-of-distribution (OOD) data where the visual patterns deviate from those seen during training. Extended analysis of dynamics and video prediction models are given in Appendix B.

We argue that explicit dynamics modeling through $p(o_{t+1}|o_t, a_t)$ is essential for robust policy learning. A key advantage of this approach is that dynamics learning can benefit from diverse trajectory data, including random explorations that require zero human annotation cost. Unlike video prediction, which requires consistent visual patterns from expert demonstrations, dynamics models can learn from any state-action-state transitions, enabling them to capture the true environmental physics across a broader state-action space. However, directly incorporating explicit dynamics into policy learning poses challenges: (1) how to leverage dynamics information during action generation, and (2) how to train dynamics and policy models effectively.

To address these challenges, we propose a ***Dynamics-Aligned Flow Matching Policy*** that integrates dynamics predictions into policy learning through iterative flow generation. Our key insight is that during flow generation, intermediate samples from dynamics and policy models can provide mutual correction to each other, leading to more accurate and generalizable policies.

Our contributions are: (1) A novel flow matching architecture that shares intermediate generated flow samples between policy and dynamics models, enabling self-correction through dynamics-predicted future observations; (2) Theoretical analysis showing improved approximation properties when the policy receives information about predicted next observations;

2. Methodology

Our approach consists of two key innovations: (1) explicit dynamics modeling using flow matching, and (2) iterative coupling between dynamics and policy generation. We present our architectural design in Section 2.2.

2.1. Notation

Let $x_t^{(\tau)}$ be a CNF of x at trajectories timestep t and flow matching timestep τ. Defining the forward process as straight paths between the data distribution $p(x_t^{(1)})$ and a

*Corresponding author, [1]Seoul National University

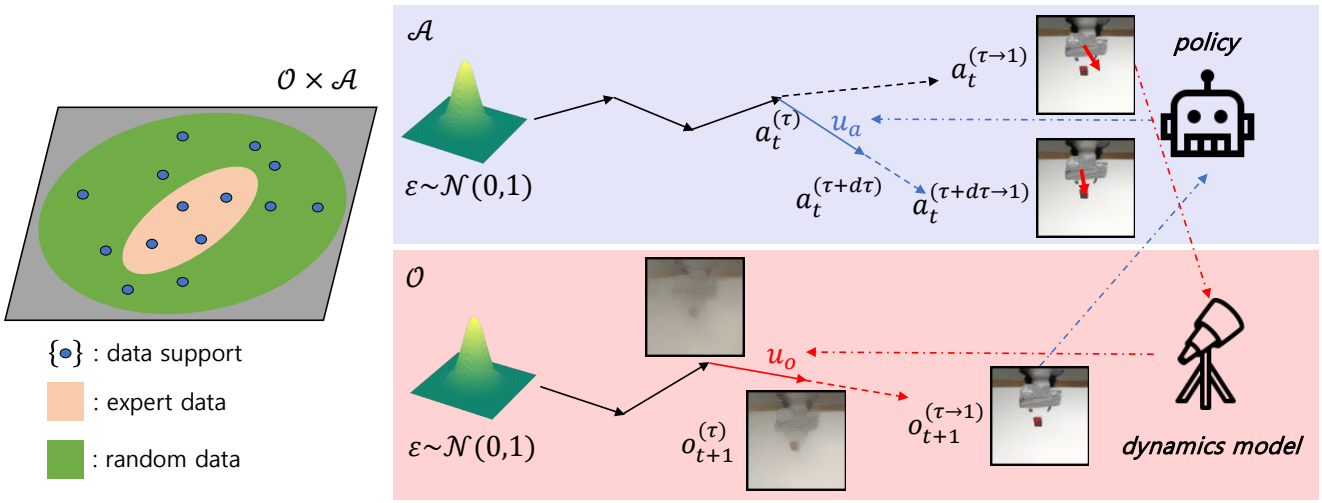

Figure 1. **(Left)** Data support comparison in observation-action space $\mathcal{O} \times \mathcal{A}$. Expert demonstrations (orange) provide limited support in the state-action space, while incorporating random trajectory data (green) significantly expands the support region, enabling better generalization beyond expert trajectories. **(Right)** Our iterative flow generation process. The dynamics model learns $p(o_{t+1}|o_t, a_t)$ from both expert and random data, while the policy model learns with future observations conditioning. During inference, intermediate flow samples from dynamics and policy models provide mutual correction through iterative generation, where $o_{t+1}^{(\tau)}$ and $a_t^{(\tau)}$ are refined at each timestep. This dynamics-aligned approach enables self-correction and improved robustness compared to standard behavior cloning methods.

Algorithm 1 Sampling

Given $o_t^{(1)}$, set $\delta\tau = 1/\text{num\_steps}$
Sample $a_t^{(0)} \sim \mathcal{N}(0, I)$, $\quad o_{t+1}^{(0)} \sim \mathcal{N}(0, I)$
for i in $1, \ldots,$ num_steps $- 1$ **do**
  $\tau = i \cdot \delta\tau$
  $a_t^{(\tau+\delta\tau)} = a_t^{(\tau)} + \delta\tau \cdot g_\phi(a_t^{(\tau)}, \tau | o_t^{(1)}, o_{t+1}^{(\tau \to 1)})$
  $o_{t+1}^{(\tau+\delta\tau)} = o_{t+1}^{(\tau)} + \delta\tau \cdot f_\theta(o_{t+1}^{(\tau)}, \tau | o_t^{(1)}, a_t^{(\tau+\to\delta\tau \to 1)})$
end for
return $o_{t+1}^{(1)}, a_t^{(1)}$

standard normal distribution $\mathcal{N}(0, I)$.

$$x_t^{(\tau)} = (1 - \tau)\epsilon + \tau x_t^{(1)} \tag{1}$$

for noise $\epsilon \sim \mathcal{N}(0, I)$ and the data point $x_t^{(1)} \sim p(x_t^{(1)})$. Also, given $x_t^{(\tau)}$, we define $x_t^{(\tau \to 1)}$ as transformation of $x_t^{(\tau)}$ to $x_t^{(1)}$ regime.

Let $\mathcal{D}_{\text{expert}} = \{(o_{t,i}^{(1)}, a_{t,i}^{(1)}, o_{t+1,i}^{(1)})\}_{i=1}^N$ denote our training dataset of expert demonstrations. Note that the superscript denotes the flow matching timestep, while the subscripts denote the timestep of trajectories and index of demonstration, respectively. We assume access to additional random trajectory data $\mathcal{D}_{\text{random}}$ collected using a random policy.

2.2. Dynamics-Aligned Flow Matching Policy

Iterative Generation Necessity. Our approach requires iterative generation due to the inherent dependency between dynamics and policy models. The dynamics model f_θ learns $(o_t, a_t) \to o_{t+1}$, requiring both current observation and action as inputs. Conversely, the policy model g_ϕ benefits from seeing the consequences of its predicted actions, necessitating predicted next observations. This mutual dependency makes simultaneous generation infeasible, motivating an iterative approach where dynamics and policy models alternate in providing feedback to each other.

Architecture Overview. Our architecture consists of two models: a dynamics model f_θ and a policy model g_ϕ. During training, we first learn the dynamics model on both expert demonstrations $\mathcal{D}_{\text{expert}}$ and random policy rollouts $\mathcal{D}_{\text{random}}$ to improve generalization. The policy is then trained with dynamics-predicted future observations as additional conditioning. Full algorithmic details are provided in Appendix C.

3. Experimental Results

	Success Rate
Flow Matching Policy	0.92/0.82
Ours	0.92/**0.85**

Table 1. Success rate of manipulation task of Robomimic `square-mh` on 22 distinct initial states. Each figure corresponds to the (maximum performance)/(average of last 5 checkpoints).

We empirically evaluate our method on the Robomimic `square-mh` benchmark as an initial validation of our approach. These preliminary experiments, conducted without extensive hyperparameter tuning, demonstrate the feasibility of dynamics-aligned flow matching. Table 1 reports the task success rates.

Acknowledgements This work is in part supported by the National Research Foundation of Korea (NRF, RS-2024-00451435(20%), RS-2024-00413957(20%)), Institute of Information & communications Technology Planning & Evaluation (IITP, RS-2021-II212068(10%), RS-2025-02305453(15%), RS-2025-02273157(15%), 2021-0-00180(10%), Artificial Intelligence Graduate School Program (Seoul National University)(10%)) grant funded by the Ministry of Science and ICT (MSIT), Institute of New Media and Communications(INMAC), the BK21 FOUR program of the Education and Research Program for Future ICT Pioneers, Seoul National University in 2025, and Samsung Electronics Co., Ltd(IO210202-08370-01).

References

[1] Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari, 2024. 5

[2] Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-based offline reinforcement learning with diversified q-ensemble, 2021. 5

[3] Kevin Black, Mitsuhiko Nakamoto, Pranav Atreya, Homer Walke, Chelsea Finn, Aviral Kumar, and Sergey Levine. Zero-shot robotic manipulation with pretrained image-editing diffusion models, 2023. 1, 5

[4] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2024. 5, 6

[5] Max Braun, Noémie Jaquier, Leonel Rozo, and Tamim Asfour. Riemannian flow matching policy for robot motion learning, 2024. 5

[6] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024. 5

[7] Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024. 1, 5

[8] Cheng Chi et al. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems (RSS)*, 2023. 1, 5, 6

[9] Sudeep Dasari, Oier Mees, Sebastian Zhao, Mohan Kumar Srirama, and Sergey Levine. The ingredients for robotic diffusion transformers. *arXiv preprint arXiv:2410.10088*, 2024. 1, 5

[10] Yilun Du, Mengjiao Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Joshua B. Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation, 2023. 1, 5

[11] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, Kyle Lacey, Alex Goodwin, Yannik Marek, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis, 2024. 6

[12] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models, 2024. 5

[13] Haoran He, Chenjia Bai, Ling Pan, Weinan Zhang, Bin Zhao, and Xuelong Li. Learning an actionable discrete diffusion policy via large-scale actionless video pre-training, 2024. 1, 5

[14] Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations, 2025. 1, 5

[15] Geon-Hyeong Kim, Seokin Seo, Jongmin Lee, Wonseok Jeon, HyeongJoo Hwang, Hongseok Yang, and Kee-Eung Kim. Demodice: Offline imitation learning with supplementary imperfect demonstrations. In *International Conference on Learning Representations*, 2021. 5

[16] Dohyeok Lee, Seungyub Han, Taehyun Cho, and Jungwoo Lee. Spqr: Controlling q-ensemble independence with spiked random model for reinforcement learning, 2024. 5

[17] Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified video action model, 2025. 5

[18] Ajay Mandlekar, Fabio Ramos, Byron Boots, Silvio Savarese, Li Fei-Fei, Animesh Garg, and Dieter Fox. Iris: Implicit reinforcement without interaction at scale for learning control from offline robot manipulation data, 2020. 5

[19] NVIDIA, :, Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, Daniel Dworakowski, Jiaojiao Fan, Michele Fenzi, Francesco Ferroni, Sanja Fidler, Dieter Fox, Songwei Ge, Yunhao Ge, Jinwei Gu, Siddharth Gururani, Ethan He, Jiahui Huang, Jacob Huffman, Pooya Jannaty, Jingyi Jin, Seung Wook Kim, Gergely Klár, Grace Lam, Shiyi Lan, Laura Leal-Taixe, Anqi Li, Zhaoshuo Li, Chen-Hsuan Lin, Tsung-Yi Lin, Huan Ling, Ming-Yu Liu, Xian Liu, Alice Luo, Qianli Ma, Hanzi Mao, Kaichun Mo, Arsalan Mousavian, Seungjun Nah, Sriharsha Niverty, David Page, Despoina Paschalidou, Zeeshan Patel, Lindsey Pavao, Morteza Ramezanali, Fitsum Reda, Xiaowei Ren, Vasanth Rao Naik Sabavat, Ed Schmerling, Stella Shi, Bartosz Stefaniak, Shitao Tang, Lyne Tchapmi, Przemek Tredak, Wei-Cheng Tseng, Jibin Varghese, Hao Wang, Haoxiang Wang, Heng Wang, Ting-Chun Wang, Fangyin Wei, Xinyue Wei, Jay Zhangjie Wu, Jiashu Xu, Wei Yang, Lin Yen-Chen, Xiaohui Zeng, Yu Zeng, Jing Zhang, Qinsheng Zhang, Yuxuan Zhang, Qingqing Zhao, and Artur Zolkowski. Cosmos world foundation model platform for physical ai, 2025. 5

[20] William Peebles and Saining Xie. Scalable diffusion models with transformers, 2023. 5

[21] Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, David Yan, Dhruv Choudhary, Dingkang Wang, Geet Sethi, Guan Pang, Haoyu Ma, Ishan

Misra, Ji Hou, Jialiang Wang, Kiran Jagadeesh, Kunpeng Li, Luxin Zhang, Mannat Singh, Mary Williamson, Matt Le, Matthew Yu, Mitesh Kumar Singh, Peizhao Zhang, Peter Vajda, Quentin Duval, Rohit Girdhar, Roshan Sumbaly, Sai Saketh Rambhatla, Sam Tsai, Samaneh Azadi, Samyak Datta, Sanyuan Chen, Sean Bell, Sharadh Ramaswamy, Shelly Sheynin, Siddharth Bhattacharya, Simran Motwani, Tao Xu, Tianhe Li, Tingbo Hou, Wei-Ning Hsu, Xi Yin, Xiaoliang Dai, Yaniv Taigman, Yaqiao Luo, Yen-Cheng Liu, Yi-Chiao Wu, Yue Zhao, Yuval Kirstain, Zecheng He, Zijian He, Albert Pumarola, Ali Thabet, Artsiom Sanakoyeu, Arun Mallya, Baishan Guo, Boris Araya, Breena Kerr, Carleigh Wood, Ce Liu, Cen Peng, Dimitry Vengertsev, Edgar Schonfeld, Elliot Blanchard, Felix Juefei-Xu, Fraylie Nord, Jeff Liang, John Hoffman, Jonas Kohler, Kaolin Fire, Karthik Sivakumar, Lawrence Chen, Licheng Yu, Luya Gao, Markos Georgopoulos, Rashel Moritz, Sara K. Sampson, Shikai Li, Simone Parmeggiani, Steve Fine, Tara Fowler, Vladan Petrovic, and Yuming Du. Movie gen: A cast of media foundation models, 2025. 5

[22] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022. 6

[23] Lirui Wang, Kevin Zhao, Chaoqi Liu, and Xinlei Chen. Learning real-world action-video dynamics with heterogeneous masked autoregression, 2025. 5

[24] Youpeng Wen, Junfan Lin, Yi Zhu, Jianhua Han, Hang Xu, Shen Zhao, and Xiaodan Liang. Vidman: Exploiting implicit dynamics from video diffusion model for effective robot manipulation. *Advances in Neural Information Processing Systems*, 37:41051–41075, 2024. 1, 5

[25] Jialong Wu, Shaofeng Yin, Ningya Feng, Xu He, Dong Li, Jianye Hao, and Mingsheng Long. ivideogpt: Interactive videogpts are scalable world models, 2024. 1, 5

[26] Minjie Zhu, Yichen Zhu, Jinming Li, Junjie Wen, Zhiyuan Xu, Ning Liu, Ran Cheng, Chaomin Shen, Yaxin Peng, Feifei Feng, and Jian Tang. Scaling diffusion policy in transformer to 1 billion parameters for robotic manipulation, 2024. 1, 5

Appendix

A. Extened Related Works

Diffusion Models for Policy Learning. Diffusion Policy [8] pioneered the use of diffusion models for robotic control, demonstrating superior performance in high-dimensional action spaces. Follow-up works [9, 26] extended this to larger scales and multi-modal inputs using Diffusion Transformer (DiT) [20]. However, these approaches rely purely on behavioral cloning without explicit dynamics modeling. More recently, π_0 [4] and RFMP [5] have adopted flow matching models for robotic policy learning, showing improved training efficiency and inference speed while maintaining comparable performance to diffusion-based approaches.

Video Prediction for Robotics. Video-based approaches for robot learning leverage the success of large-scale video prediction models [6, 19, 21] to learn implicit dynamics representations. GR-2 [7], VPDD [13], SuSIE [3], and iVideoGPT [25] use pre-trained video diffusion models for policy learning. Specifically, UniPi [10] generates next observations to train policies in an inverse dynamics operator manner, while VPP [14] and VidMan [24] leverage learned representations from video prediction models for policy training. Despite their success, these methods fundamentally learn $p(o_{t+1}|o_t)$ without explicit action conditioning, potentially missing fine-grained dynamics details. Inspired by using random data [15] and randomness in RL [2, 16] for generalization, we explicitly train dynamics model $p(o_{t+1}|o_t, a_t)$ with random and expert data.

Model-Based Methods. Traditional model-based RL learns explicit dynamics models for planning and imagination-based policy improvement. Recent works like DreamerV3 [12], IRIS [18], and DIAMOND [1] demonstrate strong performance through learned world models. However, these approaches often focus on imagination-based policy learning rather than direct integration of dynamics information into action generation.

UVA [17] and HMA [23] explore learning explicit dynamics in multimodal settings and diverse training modes, bridging large-scale video understanding with dynamics modeling. While these methods jointly train action and observation prediction heads within a unified architecture, they maintain separate generation pathways during inference. In contrast, our approach introduces iterative flow sharing during the generation process itself, where intermediate samples from dynamics and policy models provide mutual correction to each other at each timestep. This fundamental difference enables self-correction based on predicted dynamics rather than relying on independently trained prediction heads.

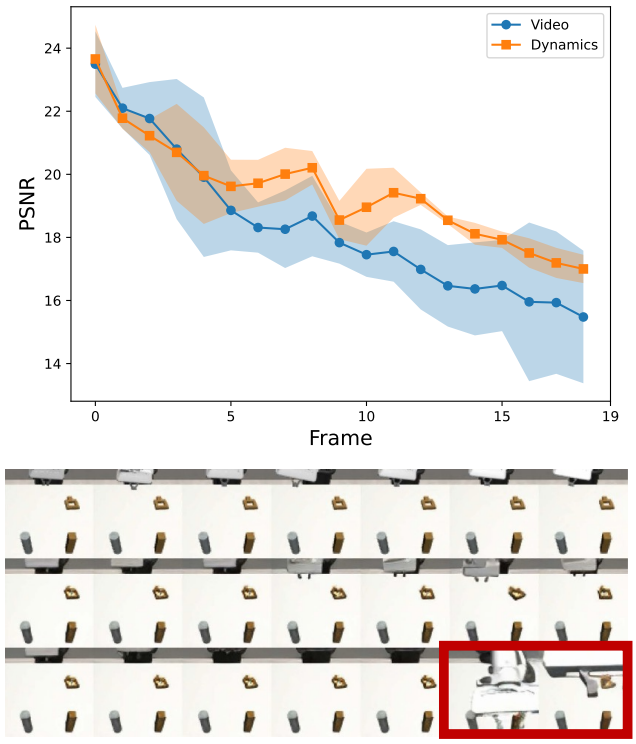

Figure 2. Comparison of PSNR curves (top) and qualitative results (bottom) for autoregressive image generation from the same initial frame. The qualitative results show ground truth, generated samples using dynamics, and video model outputs, arranged vertically.

B. Comparison of dynamics model and video prediction model

To compare our dynamics model against a pure video-prediction baseline, we trained both architectures under identical settings on the Robomimic `square-mh` dataset, differing only in whether the action embedding is provided as an input. For evaluation, we collected a held-out test set by rolling out a random policy in the `NutAssemblySquare` environment using random seeds distinct from those used in training dataset. Each model was then tasked with iteratively predicting the next observation, using its own previous prediction as input at each time step. Figure 2 summarizes this comparison, reporting PSNR curves alongside sample rollouts for both the dynamics and video-prediction models.

Figure 2 plots the PSNR of the two models' autoregressive rollouts and reveals that the dynamics model achieves consistently higher PSNR than the video-prediction baseline—indeed, the gap is visually substantial. This quantitative gap indicates that conditioning on both the current observation and the action embedding renders the dynamics model markedly more robust for multi-step future prediction.

Figure 2 shows example rollouts from each model. The video-prediction model—being purely image-

Algorithm 2 Training

Input: expert demo $\mathcal{D}_{\text{expert}}$, random demo $\mathcal{D}_{\text{random}}$

while not converged **do** ▷ Train dynamics model
  Sample $(o_t^{(1)}, a_t^{(1)}, o_{t+1}^{(1)}) \sim \mathcal{D}_{\text{expert}} \cup \mathcal{D}_{\text{random}}$
  Sample $\epsilon_o \sim \mathcal{N}(0, I),\ \epsilon_a \sim \mathcal{N}(0, I)$
  Calculate $\mathcal{L}_{\text{dyn}}(\theta)$ using Equation 2
  $\theta \leftarrow \theta - \gamma_1 \nabla_\theta \mathcal{L}_{\text{dyn}}(\theta)$
end while
while not converged **do** ▷ Train policy model
  Sample $(o_t^{(1)}, a_t^{(1)}, o_{t+1}^{(1)}) \sim \mathcal{D}_{\text{expert}}$
  Sample $\epsilon_o \sim \mathcal{N}(0, I),\ \epsilon_a \sim \mathcal{N}(0, I)$
  Calculate $\mathcal{L}_{\pi}(\phi)$ using Equation 3
  $\phi \leftarrow \phi - \gamma_2 \nabla_\phi \mathcal{L}_{\pi}(\phi)$
end while

conditioned—goes down the irrelevant trajectory. This is because small reconstruction errors compound at every step, and before long the model generates frames that have little to do with ground truth. By contrast, our dynamics model is action-conditioned at each timestep, which anchors its predictions to the actual control inputs. The action signal keeps the rollout aligned with the ground-truth trajectory, preventing the runaway "drift" seen in the video-only baseline. The consistently better PSNR performance of the dynamics model indicates that explicit dynamics modeling requires architectural inductive biases beyond a simple sequence extension, supporting our motivation for a dedicated dynamics prediction module.

C. Implementation Details

C.1. Training Objective

We first train the dynamics model with the following objective: given $\epsilon_o \sim \mathcal{N}(0, I)$, $\tau_o \sim p(\tau_o)$, $(o_t^{(1)}, a_t^{(1)}, o_{t+1}^{(1)}) \sim D = D_{\text{expert}} \cup D_{\text{random}}$,

$$\mathcal{L}_{\text{dyn}}(\theta) = \|u_o - f_\theta(o_{t+1}^{(\tau_o)}, \tau_o | o_t^{(1)}, a_t^{(1)})\|^2 \tag{2}$$

where $u_o = o_{t+1}^{(1)} - \epsilon_o$, $o_{t+1}^{(\tau_o)} = \tau_o o_{t+1}^{(1)} + (1 - \tau_o)\epsilon_o$

Then, with the dynamics model frozen, we train the policy with: given $\epsilon_a \sim \mathcal{N}(0, I)$, $\tau_a \sim p(\tau_a)$, $(o_t^{(1)}, a_t^{(1)}, o_{t+1}^{(1)}) \sim D_{\text{expert}}$,

$$\mathcal{L}_{\pi}(\phi) = \|u_a - g_\phi(a_t^{(\tau_a)}, \tau_a | o_t^{(1)}, o_{t+1}^{(1)})\|^2 \tag{3}$$

where $u_a = a_t^{(1)} - \epsilon_a$, $a_t^{(\tau_a)} = \tau_a a_t^{(1)} + (1 - \tau_a)\epsilon_a$

C.2. Training Details

As mentioned before, we used flow matching to train policy and dynamics model. We sampled the timestep τ from $\beta(1.5, 1)$ for policy, while LogitNormal(0.0, 1.0) is used for dynamics. We chose these distributions referring to prior works [4, 11]. Table 2 and 3 show the training details for both models. For training and sampling, we used the algorithms 2, 1.

C.3. Model Architecture

This section discusses the detailed model architectures. For dynamics model, we used DiT. DiT has shown scalability and strong performance in diffusion-based generative modeling. Also, for latent diffusion, we used VAE trained by [22]. In addition, we embedded the low dimension action with 2 layer MLP with GeLU activation. For policy model, we used U-Net architecture. To inject the observation information to the network, we used cross-attention layer in between residual blocks of U-Net. The overall architecture resembles to the U-Net used by [22]. We used ResNet18 for vision feature extractor of policy from scratch and trained with the policy as reported by [8]. The Table 2 and 3 show the hyperparameters of both policy and dynamics models.

# of residual blocks for each layers	2
Dimension of Head	32
Transformer Depth	1
Channel Multipliers	1, 2, 4
Attention Resolution	1, 2, 4
Channels	224
Model Parameters	217M
Flow matching timestep distribution	$\beta(1.5, 1)$
Learning rate	1e-4
Batch size	128
Epoch	200

Table 2. Hyperparameters for policy architecture(up) and training(down)

Path size	2
Hidden dimension	1152
Depth	28
Action Embedding Dimension	64
MLP ratio	4.0
Model Parameters	674M
Flow matching timestep distribution	LogitNormal(0.0, 1.0)
Learning rate	1e-4
Batch size	128
Precision	`torch.FP16`
Epoch	70

Table 3. Hyperparameters for dynamics model architecture(up) and training(down)

C.4. Computational Overhead of Proposed Policy

	Inference time [s]
Diffusion Policy-C	0.70 ± 0.10
Flow Matching Policy	0.06 ± 0.10
Ours	0.34 ± 0.01

Table 4. Sampling time for the baselines (Diffusion Policy-C, Flow Matching Policy) and ours.

When augmenting the policy with next-observation conditioning, one natural concern is the additional computational and training overhead. To quantify this, we measure: (1) inference latency, the time required to sample an action conditioned on the current observation, and (2) training overhead, the total cost to learn the dynamics model

Table 4 shows that our method introduces a clear inference-time overhead compared to the pure flow-matching policy (0.34[sec] vs. 0.06[sec]), due to the incorporation of a learned dynamics model. Nevertheless, it remains substantially faster than Diffusion Policy-C, reducing inference time by more than 50% (0.34[sec] vs. 0.70[sec]). Given that Diffusion Policy-C has already demonstrated real-time performance in a variety of robotic manipulation tasks, our lower latency strongly suggests that the proposed policy is also suitable for real-world deployment—even in settings that demand high control frequency.

Moreover, the offline training burden for the dynamics model is minimal. We train it on rollouts from a random policy—requiring no human supervision—and exploit the efficiency of flow matching to achieve convergence in under 100 epochs on all Robomimic benchmarks. In practice, the total compute added by dynamics learning is small relative to the cost of the primary policy optimization, demonstrating that our method imposes only a modest overhead while delivering substantial robustness gains.