One-Step Action Diffusion for Robotic Manipulation via Inductive Moment Matching

Emre Adabag[1] Ishaan Mahajan[1] Alexander Du[1]
[1]Columbia University {ea2944, iam2141, asd2192}@columbia.edu

***Abstract*—Visuomotor policy learning has shown strong generative capabilities for complex action distributions, but diffusion models remain computationally expensive at inference time, posing challenges for real-time applications. We present the first application of *Inductive Moment Matching* (IMM) to robotic action diffusion and demonstrate that it enables *single-step* action generation while preserving performance. Using the vision-based Push-T manipulation task, we benchmark IMM against conventional noise-prediction diffusion (DDPM/DDIM) and Flow Matching (FM) across training epochs and inference steps. Our results show that IMM achieves a normalized score of 0.6 in just a single denoising step after 100 training epochs, while both DDPM/DDIM and FM completely fail (0.0) in the one-step setting. Even with minimal training (20 epochs), IMM attains a 0.3 score in one step, when other methods show no success. Furthermore, IMM maintains consistent performance across different step counts, with scores between 0.6-0.8 at higher training epochs, demonstrating its robustness. This allows IMM to attain comparable success with 32× fewer inference evaluations, making it particularly well-suited for resource-constrained robotic platforms where rapid decision-making is critical.**

I. Introduction

Robotic manipulation tasks increasingly rely on generative models capable of effectively mapping rich sensory inputs, such as visual observations, to precise sequences of actions. Among these generative approaches, Denoising Diffusion Probabilistic Models (DDPMs) have demonstrated state-of-the-art performance in diverse domains ranging from high-quality image synthesis to visuomotor control tasks [3]. Despite their effectiveness, a significant limitation of DDPMs is their reliance on iterative inference methods. Such methods typically require tens to hundreds of denoising steps to produce accurate outputs, which poses substantial challenges for real-time control in robotics applications, particularly on embedded and compute-limited platforms.

Recent advances like Flow Matching (FM) [9, 2] have partially addressed this computational bottleneck by significantly reducing the required inference steps. FM methods provide improved computational efficiency by solving simplified Ordinary Differential Equations (ODEs), yet they still rely on multi-step solvers, necessitating multiple inference iterations.

In contrast, Inductive Moment Matching (IMM) [20] provides a principled framework for training generative models that support one- or few-step inference without the need for iterative denoising. Rather than explicitly predicting noise or velocity fields, IMM learns mappings between different noise levels by aligning full marginal distributions using moment-matching objectives such as Maximum Mean Discrepancy (MMD). This enables IMM to generate samples in a single forward pass while guaranteeing convergence to the target distribution under a self-consistent interpolant framework. While IMM has demonstrated state-of-the-art performance in image generation tasks, its application to robotic policy learning—particularly for efficient visuomotor control—has not been previously studied.

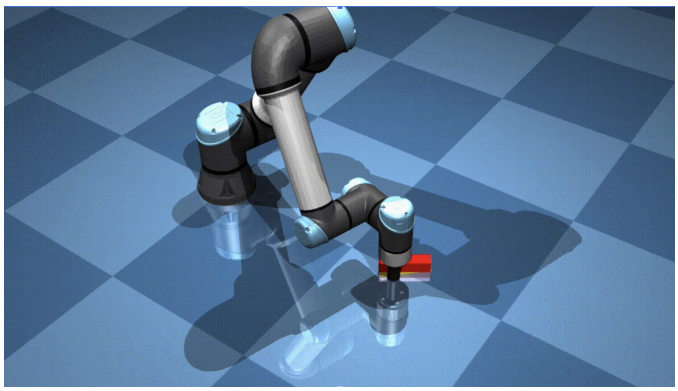

Fig. 1. Push-T Task using IMM

Motivated by this gap, our work explores the feasibility of using IMM for closed-loop robotic action generation. We evaluate whether IMM can match or exceed the performance of established baselines such as DDPM/DDIM and Flow Matching (FM), while requiring significantly fewer inference steps. Our experiments focus on the vision-based Push-T benchmark, a manipulation task requiring accurate and reactive control based solely on visual input.

Our primary contributions in this study are as follows:

- We propose the first IMM-based diffusion policy specifically designed for robotic action generation tasks.
- We conduct a thorough empirical comparison of IMM, FM, and DDPM/DDIM, carefully maintaining identical training budgets and experimental conditions to ensure fair evaluation.
- We demonstrate that IMM not only achieves superior or comparable task performance relative to DDPM/DDIM and FM but also uniquely requires only a **single** inference evaluation. This characteristic positions IMM as an exceptionally suitable approach for real-time robotic applications, particularly those constrained by computational resources.

Through these contributions, our work aims to broaden

the applicability of efficient generative modeling techniques within robotics.

II. Related Work

A. Policy Learning

Robot manipulation policies must effectively encode complex relationships between visual observations and high-dimensional action spaces while handling challenges such as multimodality, temporal consistency, and precision requirements. Prior approaches to policy learning can be broadly categorized into explicit and implicit policies.

Explicit policies directly map observations to action distributions using various parametrizations. Recent works have explored mixture density networks [12] to capture multimodality and discretized action spaces [15]. However, these approaches often struggle with accurately representing complex multimodal distributions in high-dimensional action spaces.

Implicit policies [4] reformulate the problem using energy-based models (EBMs):

$$p_\theta(\mathbf{a}|\mathbf{o}) = \frac{e^{-E_\theta(\mathbf{o},\mathbf{a})}}{Z(\mathbf{o},\theta)}$$

where $Z(\mathbf{o},\theta)$ is the intractable normalization constant. While theoretically capable of representing arbitrary distributions, EBMs in practice suffer from training instability due to the requirement for negative sampling to estimate the normalization constant.

B. Diffusion Policy

Diffusion Policy [3] addresses these challenges by leveraging a DDPM as the policy representation. Instead of directly outputting actions, the policy infers the action-score gradient to generate action sequences by an iterative denoising process. Specifically, Diffusion Policy formulates a distribution $p(\mathbf{A}_t|\mathbf{O}_t)$ of future actions conditioned on visual observations. This bypasses the challenges of estimating normalization constants by learning the score function [1], providing stable training while maintaining distributional expressivity. The authors highlight the following details to explain Diffusion Policy's impressive performance:

- **Closed-loop action sequences**: Combining high-dimensional action sequence prediction with receding-horizon control to balance long-horizon planning with reactivity.
- **Visual conditioning**: Treating visual observations as conditioning rather than part of the joint distribution, enabling efficient inference.
- **Architecture innovations**: Specialized network architectures like the time-series diffusion transformer to handle complex temporal dependencies.

Other works have also shown successful application of diffusion models in the context of robot learning; concurrently, [13, 14, 6] explored classifier-free guidance for goal-conditioned policies, efficient sampling strategies, and integration with reinforcement learning methods.

Recently, [19] extends Diffusion Policy to more complex 3D observation representations, [11] introduces a hierarchical scheme for long-horizon planning, and [5] deploy Diffusion Policy for mobile manipulation on a quadraped, even demonstrating zero-shot cross-embodiment of their policy. Despite these advances, diffusion-based policies still face challenges in inference efficiency, often requiring multiple denoising steps to generate high-quality actions. This limitation not only affects real-time control scenarios, it becomes a bottleneck wherever DDPM is used in robotics, motivating the exploration of more techniques that maintain the expressive power of diffusion models while lowering inference time.

III. Methods

To enable our research, we benchmark across three different methods.

A. Diffusion Policy

Following [3], visuomotor policy is formulated as a Denoising Diffusion Probabilistic Model (DDPM) [7]. First a forward noising process is used to transform data samples $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ into Gaussian noise $\mathbf{x}_T \sim \mathcal{N}(0,\mathbf{I})$ through a sequence of Markovian transitions. The DDPM reverses this trajectory, starting from $\mathbf{x}_T \sim \mathcal{N}(0,\mathbf{I})$ and iteratively denoising to produce $\mathbf{x}_0$:

$$\mathbf{x}_{k-1} = \alpha(\mathbf{x}_k - \gamma\epsilon_\theta(\mathbf{x}_k, k)) + \mathcal{N}(0,\sigma^2\mathbf{I})$$

where ϵ_θ is a learned noise prediction network, and α, γ, and σ are hyperparameters controlled by a noise schedule. This formulation can be interpreted as a discretization of Langevin dynamics [18], effectively performing a gradient descent step on an energy landscape [16]:

$$\mathbf{x}' = \mathbf{x} - \gamma\nabla E(\mathbf{x})$$

where $\epsilon_\theta(\mathbf{x}, k)$ approximates $\nabla E(\mathbf{x})$, the gradient of the energy function. This view connects DDPMs to score-based generative models and enables theoretical analysis of their convergence properties.

For visuomotor policy learning, where the model must predict a sequence of actions based on recent observations (images), the standard DDPM is modified to approximate the conditional distribution $p(\mathbf{A}_t|\mathbf{O}_t)$ where $\mathbf{A}_t$ represents the action trajectory and $\mathbf{O}_t$ represents the observation. The denoising process is thus:

$$\mathbf{A}_{k-1,t} = \alpha(\mathbf{A}_{k,t} - \gamma\epsilon_\theta(\mathbf{O}_t, \mathbf{A}_{k,t}, k) + \mathcal{N}(0,\sigma^2\mathbf{I}))$$

The training loss is similarly adjusted to:

$$L = \text{MSE}(\epsilon_k, \epsilon_\theta(\mathbf{O}_t, \mathbf{A}_{0,t} + \epsilon_k, k))$$

In the robotics context, this means:

- The model observes the current state (e.g., robot position, visual input)

- It generates actions by iteratively denoising from random noise
- Each denoising step requires a forward pass through the neural network
- The gradient field guides the denoising process toward high-probability action trajectories conditioned on the current observation

One key limitation is that diffusion policy requires multiple iterations (typically 10-100 steps) to generate high-quality actions, imposing significant computational overhead during deployment.

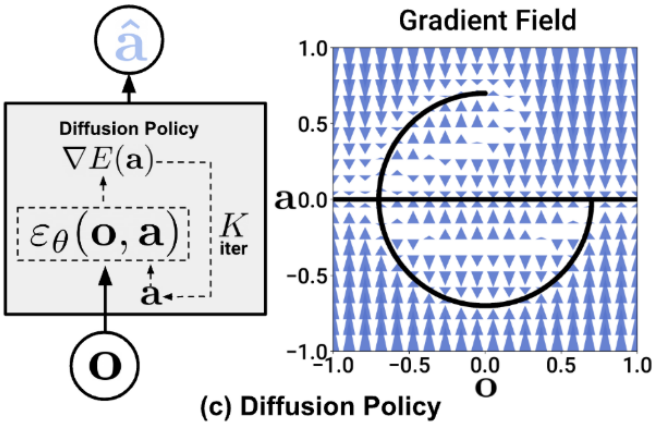

Fig. 2. Diffusion policy architecture showing the gradient field during the denoising process. The model takes observations and noisy actions as input and predicts the denoising direction, effectively creating a gradient field that guides random noise toward meaningful action distributions.

B. Flow Matching

Flow Matching [9] is deeply rooted in Optimal Transport (OT) theory, which provides a mathematical framework for finding the most efficient way to transform one probability distribution into another. The Optimal Transport problem seeks to find a mapping that minimizes the cost of moving mass from a source distribution to a target distribution. In the context of generative modeling, this translates to finding the most efficient path between a simple prior distribution (e.g., Gaussian noise) and the complex data distribution. Unlike DDPMs which focus on reversing a noising process, flow matching directly parameterizes and learns the vector field (conditional velocity) that guides the transformation between distributions:

$$\mathbf{v}_t = \alpha_t' \mathbf{x} + \sigma_t' \epsilon$$

where α_t and σ_t define an interpolation between data and prior. Flow matching trains a neural network to match these conditional velocities, which can then be integrated via a probability flow ODE to generate samples. The Flow Matching loss is defined as:

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{t, p_t(x)} \| v_t(x) - u_\theta(x, t) \|^2$$

where $v_t(x)$ is the ground truth velocity field at time t and position x, and $u_\theta(x, t)$ is the predicted velocity from the neural network. This approach allows for more direct optimization of the transport path between distributions, following OT principles of minimizing the transportation cost.

Flow Matching can be viewed as learning a continuous normalizing flow governed by an ordinary differential equation (ODE):

$$\frac{dx(t)}{dt} = v_\theta(x(t), t)$$

During inference, this ODE is solved using a numerical integrator (like Euler or Runge-Kutta methods) to transform noise samples into data samples. The OT formulation leads to straighter, more direct paths between the noise and data distributions compared to traditional diffusion paths, as visualized in Figure 3.

The main advantages of Flow Matching for robotics include:

- More flexible sampling trajectories between noise and target distributions
- Compatibility with higher-order numerical solvers (like RK4) that can take larger steps
- Ability to generate reasonable outputs with fewer steps (4-16 vs. 10-100)
- More direct paths between distributions due to the OT formulation, leading to potentially more efficient sampling

While Flow Matching reduces the computational requirements compared to standard diffusion, it still requires multiple inference steps. Also to note is that the generated actions depend on both the accuracy of the learned velocity field and the precision of the numerical integration, creating a trade-off between step count and action quality.

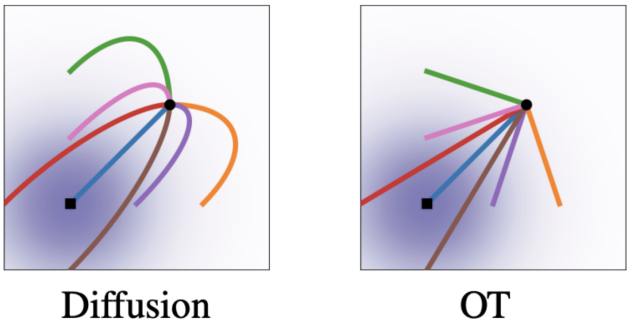

Fig. 3. Flow Matching visualization showing the differences in trajectory paths between diffusion (left) and optimal transport (right) approaches. Flow Matching enables more direct paths between distributions, following OT principles.

C. Inductive Moment Matching for Action Generation

We adopt the stochastic interpolant formulation of IMM [20], and represent action sequences as mixtures of data $\mathbf{a}$ and noise ϵ under a continuous-time interpolant:

$$q_t(\mathbf{A}_t) = \iint q_t(\mathbf{A}_t|\mathbf{a},\epsilon)q(\mathbf{a})p(\epsilon)\, d\mathbf{a}d\epsilon$$

This is a continuous spectrum of action distributions, ranging from the pure data distribution at t=0 to complete noise at t=1. To enable conditional generation, we learn a mapping from a noisy action $\mathbf{A}_t$ at time t to a less noisy action $\mathbf{A}_s$ at an earlier time $s < t$, conditioned on the observation $\mathbf{O}$:

$$p^{\theta}_{s|t}(\mathbf{A}s|\mathbf{O}) = \iint qs|t(\mathbf{A}_s|\mathbf{a},\mathbf{A}t)\, p^{\theta}s|t(\mathbf{a}|\mathbf{A}_t,\mathbf{O})\, q_t(\mathbf{A}_t)\, d\mathbf{A}_t d\mathbf{a}$$

Here, we use the DDIM interpolant [16], which ensures consistent backward mappings, for the interpolant $q_{s|t}$ between $\mathbf{a}$ and $\mathbf{A_t}$.

The model is trained using the inductive bootstrapping approach proposed in IMM. For time points $s < r < t$, two distributions are formed at time s by running a one-step process from times r and t, then, their divergence is minimized using a Maximum Mean Discrepancy (MMD) loss:

$$\mathcal{L}(\theta) = \mathbb{E}_{s,t}\left[w(s,t)\,\mathrm{MMD}^2\left(p^{\theta_-}_{s|r}(\mathbf{A}_s|\mathbf{O}),\ p^{\theta}_{s|t}(\mathbf{A}_s|\mathbf{O})\right)\right] \tag{1}$$

Where $r = r(s,t)$ is a well-conditioned mapping function with $s \le r(s,t) \le t$, θ_- denotes stop-gradient parameters, and $w(s,t)$ is a weighing function. For practicality, we use the Laplace kernel as a simplification and define the model transformation as $f^{\theta}_{s,t}(\mathbf{A}_t)$, yielding:

$$\mathcal{L}_{\mathrm{IMM}}(\theta) = \mathbb{E}\left[w(s,t)\begin{pmatrix} k(f^{\theta}_{s,t}(\mathbf{A}_t), f^{\theta}_{s,t}(\mathbf{A}'_t))+ \\ k(f^{\theta-}_{s,r}(\mathbf{A}_r), f^{\theta-}_{s,r}(\mathbf{A}'_r))- \\ k(f^{\theta}_{s,t}(\mathbf{A}_t), f^{\theta-}_{s,r}(\mathbf{A}'_r))- \\ k(f^{\theta}_{s,t}(\mathbf{A}'_t), f^{\theta-}_{s,r}(\mathbf{A}_r)) \end{pmatrix}\right] \tag{2}$$

where $f^{\theta}_{s,t}(\mathbf{A}'_t)$ is the action at time s, which is obtained by applying our model to the action at time t.

IV. Experiments

We empirically evaluate the effectiveness of IMM for vision-based robotic manipulation on the Push-T task. Our evaluation is designed to answer two core questions:

1) Can IMM achieve competitive manipulation performance using a single inference step?
2) How does IMM compare to DDPM/DDIM and Flow Matching (FM) when training and inference compute are normalized?

A. Task Setup

a) Environment: We conduct experiments in a simulated Push-T environment, first implemented in `pymunk` and later validated in `MuJoCo` [17]. In each episode, the robot must push a block from a randomized initial location to a fixed goal zone using visual observations alone. Each episode runs for up to 350 environment steps, terminating early if 95% goal coverage is achieved.

b) Observation and Action Spaces.: The agent receives 96×96 RGB image observations at each timestep, with no domain randomization or data augmentation applied. The action space consists of low-level Cartesian positions over a fixed horizon, predicted as a sequence of 16 future actions. At execution time, only the first 8 actions are applied before replanning.

B. Evaluation Protocol

Policies are deployed in a receding horizon control loop: at each step, the agent encodes the latest observation history, predicts a full 16-step action plan, executes the first 8 actions, then re-queries the policy. This setup balances long-horizon planning with short-horizon reactivity and enables robust closed-loop behavior.

We report **coverage at early exit** as our primary performance metric, defined as the percentage of the goal region covered by the object when the episode ends. Additionally, we assess:

- **Inference step efficiency:** Performance as a function of the number of test-time sampling steps $\{1, 2, 4, 8, 16, 32\}$.
- **Training compute efficiency:** Performance as a function of FLOP-normalized training epochs (EEPOCHs) $\{20, 40, 60, 80, 100\}$.

All metrics are averaged over 100 evaluation episodes with randomized initial conditions.

IMM requires two forward passes (one with gradients, one without gradients) and one backward pass per training iteration, amounting to a $1.33\times$ increase in per-step compute compared to standard DDPM and FM. To ensure a fair comparison, we normalize all evaluations in terms of *Equivalent Epochs (EEPOCHs)*. An EEPOCH is defined as one unit of compute-equivalent training. Thus, IMM models are trained for $0.75\times$ the number of wall-clock epochs relative to baselines. For instance, at 60 EEPOCHs, IMM models are trained for 45 epochs, while DDPM/FM models are trained for the full 60.

C. Baselines and Model Configuration

All methods use an identical architecture: a ResNet-18 visual encoder (with GroupNorm) followed by a 1D temporal convolutional policy network with FiLM conditioning on the observation features. No domain randomization, augmentation, or auxiliary losses were used.

We compare the following methods:

- **IMM (Ours):** A single-step generative model trained via moment matching using the DDIM interpolant. Actions

are generated in one or more direct transitions without iterative denoising.

- **DDPM/DDIM:** Standard diffusion models trained with MSE loss and evaluated with both stochastic (DDPM) and deterministic (DDIM) sampling.
- **Flow Matching (FM):** A transport-based model trained to match conditional velocity fields along a continuous interpolant using an ODE solver for sampling.

During inference, we test each model using $\{1, 2, 4, 8, 16, 32\}$ sampling steps. IMM is evaluated primarily in the single-step setting but can also operate in multi-step mode via intermediate interpolants. For FM and DDPM/DDIM, we use standard sampling schedules; FM trajectories are integrated using fixed-step Euler integration.

D. Implementation Details

All models are trained using the AdamW optimizer with a cosine learning rate decay schedule and a batch size of 64. Training is conducted on a single consumer-grade Nvidia GPU [10].

V. Results

Figures 4 and 5 break down model performance at two training budgets—20 and 100 Equivalent Epochs (EEPOCHs)—while Figure 6 provides a full heatmap of results across all training and inference step settings.

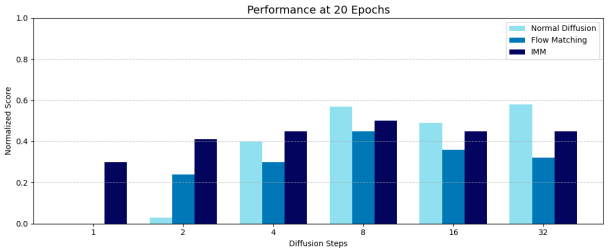

Fig. 4. Performance at 20 EEPOCHs across inference steps. IMM shows early success in one step, unlike DDPM and FM.

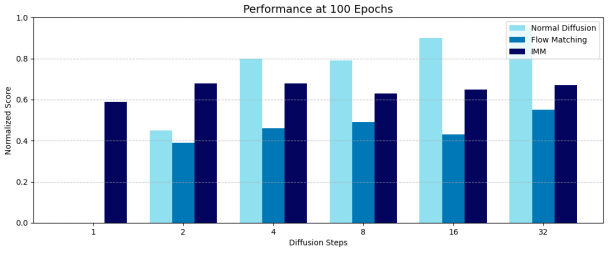

Fig. 5. Performance at 100 EEPOCHs across inference steps. IMM maintains top performance with minimal steps.

A. IMM Enables Effective Single-Step Inference

IMM consistently outperforms DDPM and FM in the one-step setting. Even with only 20 EEPOCHs of training, IMM achieves a normalized score of 0.3, while DDPM and FM both fail to register any success (0.0). At 100 EEPOCHs, IMM reaches 0.6 in a single step, still with the competing methods remaining at 0.0. This demonstrates IMM's unique ability to converge to meaningful action policies in a single inference evaluation, making it highly suited for low-latency robotic applications.

B. Robustness Across Inference Steps

Whereas FM and DDPM improve gradually with additional inference steps, IMM achieves competitive performance even with only 2–4 steps and shows only marginal gains with more. At 100 EEPOCHs, IMM achieves scores of 0.6–0.7 using 2 to 32 steps, showing little variance across this range. Notably, performance saturates or slightly decreases beyond 8 steps—suggesting that IMM effectively captures the target distribution in just a few forward passes. This is consistent with observations in prior image synthesis work [20], where IMM demonstrated strong convergence characteristics even with few steps.

However, we observe that IMM's task performance under 32 diffusion steps at 100 EEPOCHs is weaker than normal diffusion. Also, where the performance of normal diffusion significantly increases with diffusion steps, IMM's does not. We leave exploring the reasons and solutions for this occurrence to future work.

C. Training Efficiency and Convergence Behavior

IMM also converges significantly faster than both DDPM and FM. At 40 EEPOCHs, IMM achieves performance comparable to DDPM and FM models trained for 80–100 EEPOCHs with 16+ inference steps. We also observe that IMM policies exhibit smoother and more goal-directed motion trajectories during evaluation—likely a byproduct of its global moment-matching objective. In contrast, DDPM and FM tend to generate less coherent trajectories in the low-step regime, often requiring many steps to refine viable action sequences.

	Noise Prediction						Flow Matching						IMM					
EEpochs / Steps	1	2	4	8	16	32	1	2	4	8	16	32	1	2	4	8	16	32
20	0.0	0.0	0.4	0.6	0.5	0.6	0.0	0.2	0.3	0.5	0.4	0.3	0.3	0.4	0.4	0.5	0.4	0.4
40	0.0	0.2	0.7	0.8	0.9	0.9	0.0	0.4	0.5	0.5	0.6	0.5	0.6	0.8	0.6	0.8	0.7	0.7
60	0.0	0.3	0.9	0.9	0.9	0.9	0.0	0.4	0.6	0.6	0.6	0.5	0.5	0.6	0.6	0.6	0.6	0.6
80	0.0	0.5	0.8	0.9	0.9	1.0	0.0	0.4	0.5	0.6	0.6	0.5	0.6	0.7	0.7	0.8	0.8	0.8
100	0.0	0.4	0.8	0.8	0.9	1.0	0.0	0.4	0.5	0.5	0.4	0.6	0.6	0.7	0.7	0.6	0.6	0.7

Fig. 6. Full performance heatmap across EEPOCHs and inference steps for all methods. IMM achieves high scores consistently with fewer steps.

In Figure 7, we show sample rollout snapshots comparing the one-step trajectories of IMM and DDPM. IMM exhibits smoother, more direct paths to the goal, while DDPM frequently fails to generate coherent movement. These behaviors reinforce the quantitative results, highlighting IMM's capacity to generalize effectively even under extreme inference constraints.

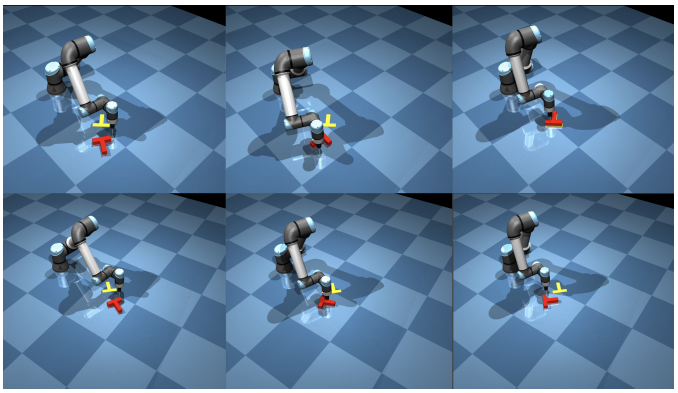

Fig. 7. Rollout snapshots comparing IMM (top) and DDPM (bottom) in single-step settings. IMM yields smooth, goal-directed trajectories; DDPM often fails to produce coherent motion.

D. Summary

IMM achieves state-of-the-art efficiency among single-stage generative models for robotic action generation. Across all tested training regimes:

- IMM is the only method with meaningful single-step performance.
- IMM maintains stable performance across different numbers of inference steps.
- IMM converges faster than baselines, reaching comparable or superior performance with fewer training epochs.

VI. Conclusion

Our results demonstrate that Inductive Moment Matching (IMM) offers a highly effective approach for action generation in robotic manipulation tasks, particularly when inference efficiency is a key constraint. IMM not only outperforms standard diffusion (DDPM/DDIM) and Flow Matching (FM) methods in the single-step setting, but it also maintains strong performance with minimal increases in inference steps. This positions IMM as a practical candidate for real-time robotic control systems, especially in resource-constrained environments such as embedded platforms or edge devices. Unlike other few-step generative models (e.g., consistency models), IMM exhibits stable convergence behavior without requiring pretraining, distillation, or extensive architectural tuning. This robustness is consistent with results in the original IMM work on image generation [20]. Our study confirms that this advantage carries over to robotics domains, where reliable optimization is particularly valuable due to limited simulation throughput or real-world data. Additionally, state-of-the-art vision-language-action (VLA) models—such as Google's OpenVLA [8] and Physical Intelligence's Pi0 [2]—have adopted diffusion-based architectures to enable flexible, multimodal policy generation. These models often require non-trivial architectural modifications, such as action chunking, parallel rollouts, or coarse-to-fine decoders, to reduce inference overhead to manageable levels. Even then, they typically run at 8–16 denoising steps per action decision. Reducing this requirement to a single step would yield an order-of-magnitude speedup in inference, enabling the deployment of larger and more expressive VLA models in latency-sensitive or compute-constrained settings.

While IMM shows impressive performance on Push-T, our evaluation is limited in several respects. First, we focus on a single robotic manipulation task with a relatively low-dimensional control space and modest scene complexity, which may not reflect the challenges of more diverse or dynamic environments. Second, IMM's reliance on MMD introduces sensitivity to kernel choice and sample variance, which may affect stability in higher-dimensional or highly multimodal distributions. Third, although IMM enables compute-efficient inference, its training cost is slightly higher per step than DDPM or FM, and may require careful tuning or batching strategies for large-scale applications. Finally, IMM currently operates as a flat policy model; it remains an open question how well it integrates with hierarchical or goal-conditioned planners in long-horizon or task-decomposed settings. Future work should explore IMM's generalization to more complex manipulation scenarios involving:

- High-dimensional control spaces, such as 6-DOF arms or legged robots.
- 3D perception, point cloud observations, or multi-camera inputs.
- Real-world deployment on embedded hardware to test runtime efficiency.

Additionally, IMM's training formulation involves slightly higher per-step compute than DDPM or FM, which may require tuning for large-scale training or deployment pipelines. Finally, it remains an open question how IMM might be integrated into hierarchical or hybrid planning schemes. For example, one could envision combining IMM with learned value functions, classical planners, or graph-based scene understanding to support longer-horizon behaviors or task-level reasoning.

References

[1] Georgios Batzolis, Jan Stanczuk, Carola-Bibiane Schönlieb, and Christian Etmann. Conditional image generation with score-based diffusion models, 2021. URL https://arxiv.org/abs/2111.13606.

[2] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. pi_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[3] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.

[4] Pete Florence, Corey Lynch, Andy Zeng, Oscar Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning, 2021. URL https://arxiv.org/abs/2109.00137.

[5] Huy Ha, Yihuai Gao, Zipeng Fu, Jie Tan, and Shuran Song. Umi on legs: Making manipulation policies mobile with manipulation-centric whole-body controllers, 2024. URL https://arxiv.org/abs/2407.10353.

[6] Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies, 2023. URL https://arxiv.org/abs/2304.10573.

[7] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. URL https://arxiv.org/abs/2006.11239.

[8] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[9] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[10] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL https://arxiv.org/abs/1711.05101.

[11] Xiao Ma, Sumit Patidar, Iain Haughton, and Stephen James. Hierarchical diffusion policy for kinematics-aware multi-task robotic manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18081–18090, 2024.

[12] 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, 2021. URL https://arxiv.org/abs/2108.03298.

[13] Tim Pearce, Tabish Rashid, Anssi Kanervisto, Dave Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, and Sam Devlin. Imitating human behaviour with diffusion models, 2023. URL https://arxiv.org/abs/2301.10677.

[14] Moritz Reuss and Rudolf Lioutikov. Multimodal diffusion transformer for learning from play. In *2nd Workshop on Language and Robot Learning: Language as Grounding*, 2023. URL https://openreview.net/forum?id=nvtxqMGpn1.

[15] Nur Muhammad Mahi Shafiullah, Zichen Jeff Cui, Ariuntuya Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning k modes with one stone, 2022. URL https://arxiv.org/abs/2206.11251.

[16] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution, 2020. URL https://arxiv.org/abs/1907.05600.

[17] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109.

[18] Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, page 681–688, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.

[19] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations, 2024. URL https://arxiv.org/abs/2403.03954.

[20] Linqi Zhou, Stefano Ermon, and Jiaming Song. Inductive moment matching. *arXiv preprint arXiv:2503.07565*, 2025.