2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Shifted Flow Policy: Uncertainty-aware Time Reparameterization for Visuomotor Learning

Dasom Ahn*,1, Chanhyuk Jung*,1, Joonki Baek[2], Sungkeun Yoo[2], Byoung Chul Ko**,1

***Abstract*— Imitation learning for robotics often uses action chunking to mitigate the compounding errors associated with autoregressive policies. By predicting multiple future actions simultaneously, action chunking limits the accumulation of errors but introduces new difficulties. In particular, it relies on outdated observations to predict future actions, which can lead to inaccuracies. In this study, we propose Shifted Flow Policy (SFP), a simple yet effective alternative to action chunking. The SFP reparameterizes time by linearly shifting the time steps for future actions, thereby capturing the natural increase in uncertainty over time. This formulation allows each predicted action to be conditioned on up-to-date observations. Experimental results on the Push-T and MimicGen benchmarks demonstrate that SFP outperforms state-of-the-art action chunking methods across a variety of manipulation tasks by achieving higher success rates and faster inference. These findings suggest that shifted flow provides a robust and practical alternative to action chunking in visuomotor policy learning. Our code is available at https://shifted-flow-policy.github.io**

I. INTRODUCTION

Recent advances in visuomotor policy learning [1–5] have enabled robots to perform complex manipulation tasks by using visual observations. These visuomotor policies aim to map visual observations to low-level control actions, thereby allowing robots to operate in unstructured environments without requiring explicit intermediate planning or symbolic reasoning. A wide range of approaches has been proposed to learn such policies, including autoregressive sequence models [6], diffusion-based generative models [1, 7, 8], and more recently, flow-based methods [9, 10].

Early methods typically adopted behavioral cloning with recurrent architectures [11] and predicted actions autoregressively from expert demonstrations. Although these methods are simple and effective for short-horizon tasks, these models are problematic because they are afflicted by the well-known *compounding error* problem. This problem, which involves the accumulation of small prediction inaccuracies over time, leads to significant deviations in long-horizon settings. *Action chunking* has been proposed to solve this issue by predicting multiple actions instead of one. This effectively reduces the horizon of the task mitigating compounding errors.

Another important aspect of imitation learning in robotics is that actions are inherently multimodal, where a single task can be completed in multiple approaches. To better capture the multimodal action distributions, implicit policies where actions are predicted by generative models such as energy-based models [12], diffusion models [1, 7, 8], and flow matching [9, 10] has been used.

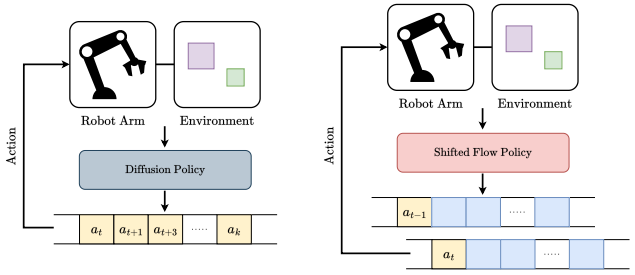


Fig. 1: Comparison between action chunking and the proposed *shifted flow*. (Left) Action chunking predicts a fixed-length sequence of actions that all share the same timestep and are executed in an open-loop manner by relying on a single observation. In contrast, (Right) shifted flow assigns linearly increasing timesteps to each action within the prediction window to model future uncertainty by enabling updated observations to condition each sampling step.

Although action chunking solves the compounding error problem it introduces new issues. The predicted actions are executed as an open-loop control system where additional feedback or observations are not utilized. This causes actions further into the future to be inaccurate causing local prediction errors. Action chunking may have been able to reduce global errors but this results in higher local errors. Several approaches have been proposed to overcome this limitation. One approach involves recomputing action chunks at every execution step to combine past and present predictions for a more accurate prediction [6]. Another line of work has improved the expressiveness of generative models by replacing diffusion models with flow-based models [9, 10]. In addition, discretization of the action space into latent tokens using the VQ-VAE [13] has been explored. These methods enhance the modeling capacity, but they do not directly address the core limitation: the reliance on outdated observations. This issue remains prevalent, as recent vision-language-action (VLA) models such as Octo [14] and π_0 [15] continue to use action chunking in their flow-based decoding heads.

Our key insight is that future actions predicted from outdated observations should be regarded as inherently unreliable, because uncertainty naturally increases with prediction horizon. To address this issue, we propose a simple time

* Equal contribution.
** Corresponding author.
[1] Department of Computer Engineering, Keimyung University, Daegu, Republic of Korea.
[2] Department of Robot Engineering, Keimyung University, Daegu, Republic of Korea.
{tommydasomahn, seagullcjung}@gmail.com, 1114830@stu.kmu.ac.kr, {skyoo, niceko}@kmu.ac.kr

reparameterization in which timesteps are linearly shifted within the prediction window rather than being kept identical for all predicted actions. This shifted timestep formulation, termed the Shifted Flow Policy (SFP), explicitly models increasing uncertainty while aligning each action with updated observations. As a result, SFP enables actions to be generated at every sampling step, unlike chunking-based methods that require multiple steps, and prevents local errors from propagating into global errors by immediately correcting deviations with fresh observations.

SFP is trained on a novel *shifted flow* constructed using these shifted time steps. First, we designed a specific shifting function that allows the model to generate action trajectories with temporally increasing uncertainty. This shifted flow replaces flow matching in the previous study with minimal architectural changes. With only minor modifications to the timestep encoder, SFP achieves improved performance compared to action-chunking methods and sets a new state-of-the-art approach across several manipulation tasks. Our method is designed to be broadly applicable and can be integrated into existing flow-based policies that rely on action chunking.

Contributions. The primary contributions of this work are summarized as follows:

- We propose shifted flow, a novel flow-based generative model that effectively replaces action chunking in visuomotor policies by introducing the concept of shifting time steps.
- We developed SFP, a visuomotor policy trained on top of the newly designed shifted flow, which serves as the underlying generative model for learning and executing action sequences. Our experiments demonstrate that SFP significantly increases sampling speed compared to existing approaches.
- We conducted extensive evaluations across a range of manipulation tasks on both the Push-T [1] and MimicGen [16] benchmarks. The results consistently show that SFP outperforms state-of-the-art action chunking baselines in terms of accuracy, generalization, and runtime efficiency.

II. RELATED WORKS

Action chunking has emerged as a key technique for scalable and robust policy learning across imitation and vision-language-action frameworks. Therefore, we focused on related studies that leverage action chunking in visuomotor policy learning.

A. Diffusion Models

Recent advances in diffusion models have revolutionized generative modeling across a variety of domains, including images [17–20], videos [21–24], audio [25–28], and 3D generation [29, 30]. Early work, such as Denoising Diffusion Probabilistic Models (DDPM) [17], employs a Markovian forward and reverse process to iteratively refine noisy samples, whereas Denoising Diffusion Implicit Models (DDIM) [18] accelerate generation by leveraging a non-Markovian sampling strategy. Building on these foundations, models such as latent diffusion models [31–33] and consistency models [34, 35] have further improved efficiency and scalability, thereby expanding the application of diffusion-based approaches to different types of data. Recently, Rolling Diffusion [19] introduced a sliding-window denoising process that captures temporal dependencies and uncertainties through progressive noise injection for video generation, enabling it to maintain temporal consistency and flexibility in sequential data.

B. Visuomotor Policies

The primary objective of imitation learning in robotics is to mimic expert demonstrations. This is typically accomplished via behavior cloning, which formulates this problem as a supervised learning problem by mapping observations to the corresponding actions. Instead of predicting a single action at each step, grouping multiple actions into chunks has been shown to be highly effective in methods such as action chunking with transformers (ACT) [6] and Diffusion Policy (DP) [1]. ACT reduces the task horizon by predicting multiple actions, thereby shortening the error accumulation chain and mitigating the compounding error issues commonly observed in autoregressive models. Because action chunks are executed sequentially without updated observations, ACT employs temporal ensembling; the model is queried multiple times, and the resulting action chunks are combined to produce the final prediction. In contrast, DP generates actions through a diffusion process conditioned on observations. Although action chunking is also used to group a subsequence of predicted actions, unlike ACT, it relies on outdated observations rather than incorporating updated inputs during inference.

Building on action chunking, VQ-BeT [36] augments behavior transformers (BeT) [37] by replacing k-means clustering with a vector-quantized variational autoencoder (VQ-VAE) [13] to tokenize action chunks into latent actions. This framework enables the VQ-BeT to more effectively capture the multimodal action distributions present in expert demonstrations. Flow matching policy (FMP) [9] adopts flow matching as an alternative to diffusion models to improve the multimodal modeling capabilities. FMP inherits the improved sampling efficiency of flow models rather than relying on the slow sampling method of diffusion-based policies. Streaming diffusion policy (SDP) [7], which preceded the use of flow models, sought to accelerate diffusion-based sampling by denoising partially noisy actions. Because the actions are reused, an action buffer is used to track the timesteps and noisy actions.

C. Vision-Language-Action Models

The advancement of visuomotor policies has resulted in the emergence of vision-language-action (VLA) models as a promising approach for training policies capable of performing multiple tasks. RT-2 [38] fine-tuned a visually conditioned language model (VLM) on Internet-scale robotics

datasets by treating robot actions as another language. Subsequently, an open-source VLA model, OpenVLA [39], was introduced. Although these models demonstrate multi-tasking capabilities across diverse robotic platforms, their autoregressive nature makes them susceptible to compounding errors. Octo [14] overcomes this problem by delegating action prediction to a diffusion-based decoder head using action chunking to mitigate compounding errors. Building on this progression, π_0 [15] further improves sampling efficiency and multimodal modeling abilities by replacing the diffusion-based decoder with a flow-based alternative.

III. METHOD

We propose shifted flow, in which the uncertainty surrounding future actions increases monotonically. Using this flow, we train a policy named Shifted Flow Policy (SFP). Unlike standard autoregressive policies that rely only on the most recent steps, SFP reparameterizes time to model uncertainty explicitly and aligns each action with updated observations. This gives rise to an efficient sampling method that can generate an infinite number of actions during the sampling process.

A. Background: Flow Matching

The goal of generative models in imitation learning is to produce action trajectories that mimic expert demonstrations. Given a dataset of demonstrations $\mathscr{D} = \{(O_i, A_i) \mid i \in \{1, \cdots, N\}\}$, where O_i is a sequence of observations and A_i is the corresponding actions, the aim is to train a model to approximate the underlying data distribution over actions conditioned on observations.

Flow matching [40] approaches this by defining a time-dependent probability path $\{p_t\}_{t\in[0,1]}$ that transforms a simple source distribution p_0, typically a standard Gaussian, into a complex target distribution p_1 such as the data distribution. This change in distribution can be defined by an ordinary differential equation (ODE) determined by a vector field whose solution is the flow which warps initial points sampled from p_0 to be distributed in p_t. The vector field can be regressed with a neural network and by solving the ODE we can sample from p_1. However, a naive regression objective, flow matching, is intractable. Conveniently, conditioning the vector field on data samples results in an objective whose gradients are equivalent to the original flow matching objective. This is called the conditional flow matching objective. A simple vector field corresponding to an affine gaussian transformation can be defined with the flow

$$\psi_t(A) = \sigma_t(A) + \mu_t(A^1), \tag{1}$$

where μ_t and σ_t is the mean and std. The conditional vector field has the form

$$u_t(A \mid A^1) = \frac{\sigma_t'(A^1)}{\sigma_t(A^1)}(A - \mu_t(A^1)) + \mu_t'(A^1). \tag{2}$$

Despite its advantages, standard flow matching assumes a static time step formulation that fails to account for the increasing uncertainty associated with predicting future actions. This motivated our reformulation of the generative process using shifted time steps, which enables a time-aware representation of uncertainty and improves the expressiveness of the learned policy.

B. Shifted Timesteps

Let $A_k \in \mathbb{R}^{W\times D}$ be a sequence of actions starting from the action index k to $k+W$. We refer to this action sequence as a window, where W is the window size, and D is the number of features of the action representation. Our key idea is to shift the time steps so that the window A_k includes time steps $t = [1, (W-1)/W, \cdots, 1/W]$. The shifting of the time step t can be expressed as

$$t_w \coloneqq \mathrm{clip}\left(t - f(w)\right), \tag{3}$$

where time step t is shifted by a monotonically increasing function $f(w)$. We clip negative values to 0 to ensure that t_w remained in the range $[0,1]$. A special case arises when time step t increases by $1/W$ and is also shifted by $1/W$:

$$t_w \coloneqq \mathrm{clip}\left(t - \frac{w}{W}\right), \tag{4}$$

$$t^i = \frac{i}{W}, \tag{5}$$

where $i \in \{0, 1, \cdots, W\}$. We can show that $t_w^i = t_{w-1}^{i-1}$ if we substitute t in Equation (4) with Equation (5) by combining the indices i and w:

$$t_w^i = \mathrm{clip}\left(\frac{i-w}{W}\right). \tag{6}$$

By further substituting i with $i-1$ and w with $w-1$, we obtain an interesting relation:

$$\begin{aligned} t_{w-1}^{i-1} &= \mathrm{clip}\left(\frac{(i-1)-(w-1)}{W}\right) \\ &= \mathrm{clip}\left(\frac{i-w}{W}\right) \\ &= t_w^i. \end{aligned} \tag{7}$$

Using this property, we can show that $t_{1:W-1}^W = t_{0:W-2}^{W-1}$. This means that, after the final sampling step, we can shift the window to the right by 1 to return to time $t_w^i = t_{0:W-2}^{W-1}$. The first action, which has completed its sampling steps, is outside the window and ready for execution. The current window lacks an action on the last index $w = W-1$. Since the time step for the last action is

$$t_{W-1}^{W-1} = \mathrm{clip}\left(\frac{(W-1)-(W-1)}{W}\right) = 0, \tag{8}$$

we can set t_{W-1}^{W-1} as noise sampled from a standard Gaussian distribution for the next sampling step. To obtain a new sample, we repeat the sampling process from $t = (W-1)/W$ to $t = 1$. The timestep t_w^W is reverted to t_w^{W-1}, and this process can be repeated infinitely, as shown in Fig. 2.

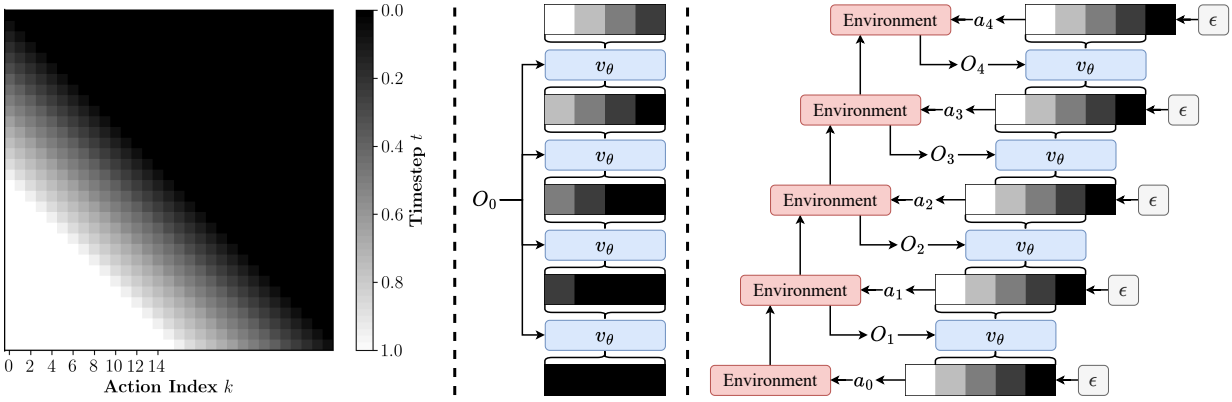


Fig. 2: **Sampling process of shifted flow. (Left) Shifted Timesteps.** The timestep t is continuously shifted as the action index k increases. By shifting t at the same rate as that at which it increases, a diagonal pattern emerges where noisy black actions are pushed towards white valid actions. **(Middle) Initlize Window.** Following the shifted timesteps t_w, the window can be initialized from noise where the final actions have timesteps $t_w = [1, (W-1)/W, \cdots, 2/W, 1/W]$. **(Right) Shifting Window.** After retrieving the first action to execute on the environment, the window shifts to the right and adds noise to the end as a new observation O_1 is observed from the environment. This process is repeated to generate more actions.

C. Shifted Flow

Consider a Gaussian conditional probability path

$$p_t(A_k|A_k^1) = \mathcal{N}(A_k|\mu_t(A_k^1), \sigma_t(A_k^1)^2 I), \tag{9}$$

where the standard Gaussian distribution at p_0 is transformed into the data distribution q at p_1. The shifted Gaussian conditional probability path can be derived by shifting the time step t according to Equation (4) such that $\mu_t(A_k) = t_w A_1$, and $\sigma_t(A_k) = 1 - t_w$. Expanding t_w shows that μ_t and σ_t can be expressed as a function of t:

$$\mu_t(A_k) = t_w A_1 = \text{clip}\left(t - \frac{w}{W}\right) A_k^1, \tag{10}$$

$$\sigma_t(A_k) = 1 - t_w = 1 - \text{clip}\left(t - \frac{w}{W}\right). \tag{11}$$

The corresponding flow has the form

$$\psi_t(x) = \sigma_{t_w}(x_1)x + \mu_{t_w}(x_1). \tag{12}$$

Substituting σ_{t_w} and μ_{t_w} with Equation (10) and Equation (11) gives us

$$\begin{aligned}\psi_t(A_k) &= (1 - t_w)A_k + t_w A_k^1 \\ &= \left(1 - \text{clip}\left(t - \frac{w}{W}\right)\right) A_k \\ &\quad + \text{clip}\left(t - \frac{w}{W}\right) A_k^1,\end{aligned} \tag{13}$$

where the flow follows a shifted Optimal Transport (OT) path. Although timestep t_w for $w \geq 1$ does not reach $t = 1$, we can reuse the flow ψ_t at time $t_{1:W-1}^W$ as $t_{0:W-2}^{W-1}$ to ensure that all actions eventually reach $t = 1$. The missing action where $t_w = t_{W-1}^{W-1}$ can be set to noise as $t_{W-1}^{W-1} = 0$. The shifted conditional vector field takes the following form:

$$u_{t_w}(A_k|A_k^1) = \frac{\sigma'_{t_w}(A_k^1)}{\sigma_{t_w}(A_k^1)}\left(A_k - \mu_{t_w}(A_k^1)\right) + \mu'_{t_w}(A_k^1). \tag{14}$$

We are mostly interested in t_w when $t - w/W > 0$ where $t_w \neq 0$. Therefore, the time derivative of t_w, $t'_w = 1$. Then $\sigma'_{t_w}(A_k^1)$ and $\mu'_{t_w}(A_k^1)$ becomes trivial to solve:

$$\mu'_{t_w}(A_k^1) = A_k^1, \tag{15}$$

$$\sigma'_{t_w}(A_k^1) = -1. \tag{16}$$

Solving for the shifted conditional vector field in closed form yields

$$u_{t_w}(A_k|A_k^1) = \frac{A_k^1 - A_k}{1 - t_w}. \tag{17}$$

Using this shifted conditional vector field, we can modify the standard CFM [40] objective as

$$\begin{aligned}&\mathscr{L}_{\text{CFM}}(\theta) = \\ &\mathbb{E}_{t_w, A_k^{t_w}, A_k^1} \|v_\theta(A_k^{t_w}, t_w, O_k) - (A_k^1 - A_k^0)\|^2,\end{aligned} \tag{18}$$

where v_θ is a neural network conditioned on observations O_k. t_w is sampled using Equation (4) with $t \sim \mathscr{U}[0,1]$. In the case of shifted flow, most of the sampling time is spent on t_w^{W-1}. Therefore, we train with another term with time t fixed to t_w^{W-1}:

$$\mathscr{L}_{\text{Window}}(\theta) = \mathbb{E}_{A_k^1} \|v_{t_w^{W-1}}(A_k^{t_w^{W-1}}) - (A_k^1 - A_k^0)\|^2. \tag{19}$$

Overall, our training loss is

$$\mathscr{L}(\theta) = \mathscr{L}_{\text{CFM}}(\theta) + \mathscr{L}_{\text{Window}}(\theta). \tag{20}$$

Figure 2 shows the sampling process for the shifted flow. First, the window is initialized from noise to a window with increasing uncertainty. Then, the first action is executed in the environment, the window is shifted, and noise is added at the end of the window. Using the updated observations, another action is sampled, and the process is repeated until the task is complete. The detailed procedure for the SFP sampling process is presented in Algorithm 1.

	Basic Tasks		Contact-Rich Tasks						Long-Horizon Tasks			
	Stack	Stack Three	Square	Threading	Coffee	Three Piece Assembly	Hammer Cleanup	Mug Cleanup	Kitchen	Nut Assembly	Pick Place	Coffee Prep
DP	$91.80^{\pm 1.41}$	$69.40^{\pm 2.99}$	$32.60^{\pm 4.64}$	$37.80^{\pm 3.74}$	$55.80^{\pm 3.16}$	$18.60^{\pm 2.60}$	$38.10^{\pm 2.90}$	$39.00^{\pm 3.10}$	$82.60^{\pm 3.42}$	$71.00^{\pm 2.70}$	$52.30^{\pm 2.91}$	$47.10^{\pm 3.04}$
SDP	$90.00^{\pm 2.15}$	$65.20^{\pm 2.73}$	$25.80^{\pm 3.06}$	$23.20^{\pm 2.94}$	$65.00^{\pm 4.61}$	$11.20^{\pm 3.12}$	$27.90^{\pm 2.10}$	$32.50^{\pm 2.05}$	$71.50^{\pm 2.97}$	$68.20^{\pm 2.40}$	$49.50^{\pm 2.63}$	$41.30^{\pm 2.30}$
FMP	$\mathbf{95.20^{\pm 1.14}}$	$\mathbf{76.40^{\pm 1.90}}$	$33.00^{\pm 2.67}$	$46.20^{\pm 2.68}$	$63.40^{\pm 2.00}$	$\mathbf{33.00^{\pm 1.94}}$	$45.40^{\pm 1.90}$	$51.20^{\pm 2.70}$	$83.00^{\pm 2.73}$	$79.50^{\pm 1.75}$	$74.20^{\pm 2.32}$	$62.50^{\pm 1.68}$
SFP	$93.00^{\pm 1.49}$	$74.00^{\pm 2.66}$	$\mathbf{41.80^{\pm 2.85}}$	$\mathbf{51.00^{\pm 1.49}}$	$\mathbf{74.40^{\pm 3.99}}$	$22.20^{\pm 2.38}$	$\mathbf{47.80^{\pm 3.01}}$	$\mathbf{59.30^{\pm 2.63}}$	$\mathbf{89.10^{\pm 1.70}}$	$\mathbf{85.32^{\pm 1.82}}$	$\mathbf{76.30^{\pm 2.28}}$	$\mathbf{72.00^{\pm 1.60}}$

TABLE I: **Quantitative results on MimicGen suite.** We repeat the evaluation 10 times and compute the average. $\pm$ indicates the 95% confidence interval.

Algorithm 1 SFP Sampling Process

Require: v_θ, W, with precomputed t_w
  $k = 0$
  Sample $A^0 \sim \mathcal{N}(0, \mathbf{I})$
  $A_k \leftarrow A^0$
  for $i = 0, 1, \cdots, W-1$ **do**
    $\Delta t \leftarrow t_w^{i+1} - t_w^i$ // window-samplings interval
    $A_k \leftarrow A_k + v_\theta(A_k, t_w^i, O_k) \cdot \Delta t$
  end for
  repeat
    $\hat{A} \leftarrow A_k[0]$
    Execute $\hat{A}$
    $k \leftarrow k+1$
    Get new observations O_k
    ▷ Shift window to the right
    $A_k[0 : W-2] \leftarrow A_k[1 : W-1]$
    Sample $A^0 \sim \mathcal{N}(0, \mathbf{I})$
    $A_k[W-1] \leftarrow A^0$
    $A_k \leftarrow A_k + v_\theta(A_k, t_w^{W-1}, O_k) \cdot \Delta t$
  until Complete

Method	Mean Score	Success Rate	Latency (ms)	Throughput
DP	$85.18^{\pm 1.31}$	$46.00^{\pm 2.99}$	368.40	21.72
SDP	$82.78^{\pm 1.37}$	$35.40^{\pm 4.99}$	104.60	76.48
FMP	$95.26^{\pm 0.51}$	$\mathbf{65.60}^{\pm 2.06}$	77.70	102.96
SFP	$\mathbf{96.43}^{\pm 0.73}$	$60.00^{\pm 5.17}$	**5.22**	**191.66**

TABLE II: **Quantitative results on Push-T.** As in the MimicGen suite we compute an average of 10 rollouts with $\pm$ indicating the 95% confidence interval. Latency is measured as an average of 50 rollouts with 10 rollouts used for warmup.

IV. EXPERIMENTS

This section attempts to answer several key research questions. First, we investigate whether shifted flow can effectively serve as a viable alternative to action chunking, a method that has proven to be highly successful in various applications. Second, we examine whether shifted flow, despite its autoregressive nature, are susceptible to the compounding errors commonly observed in such models. Finally, we explore the robustness of the shifted flow in real-world scenarios, considering that the absence of action chunking often results in unstable or oscillatory behavior in robotic control.

A. Experimental Setup

We chose the Push-T [1] dataset as a simple toy problem for examining specific parts of our method. Although the task itself is simple, it requires modeling contact-rich dynamics to generalize to new settings. For general performance comparisons, we evaluated each baseline and our method on the MimicGen dataset [16], which contains a diverse suite of tasks, including basic, contact-rich, and long-horizon tasks. Although several datasets with language task specifications exist whose scene settings are more complex, the task itself is similar in difficulty. Therefore it is redundant to evaluate on these datasets. Each method was trained for 3000 epochs using the same observation encoder and optimizer with the best hyperparameters for each method. The model architecture is relatively similar, with the only difference being the timestep encoder modified to support vectors instead of scalars. We compute the mean scores and success rates for Push-T and compute success rates for MimicGen with a total of 10 runs to reduce stochasticity.

1) Baselines: We compared our method with three representative baselines selected on the basis of their use of action chunking and similarity to our method:

- **Diffusion Policy (DP)** [1] diffusion models to generate action chunks.
- **Streaming Diffusion Policy (SDP)** [7], similar to DP, predicts action chunks with partially noisy actions tracked with an action buffer.
- **Flow Matching Policy (FMP)** [9] is a state-of-the-art visuomotor policy that leverages flow matching for improved multimodal action modeling.

B. Comparisons with Action Chunking

Table I presents the quantitative evaluations of the MimicGen benchmark, covering a range of manipulation tasks categorized into basic, contact-rich, and long-horizon settings. Overall, the results demonstrate that SFP consistently outperforms existing approaches that rely on action chunking. For basic tasks, although FMP achieved the highest success rates, SFP delivered comparable performance, indicating that the proposed formulation maintained high reliability, even in straightforward settings. However, the performance gap becomes more pronounced in contact-rich tasks, in which physical interactions introduce greater variability. In these tasks, SFP surpasses both diffusion-based (DP, SDP) and flow-based (FMP) methods but also demonstrates stronger generalization to complex dynamics. Notably, on long-

horizon tasks, which are the most challenging owing to their extended planning requirements, the SFP exhibits the most significant improvements. Structured temporal modeling enabled by shifted time steps allows the policy to more effectively manage uncertainty and execute multistage manipulation without degrading performance.

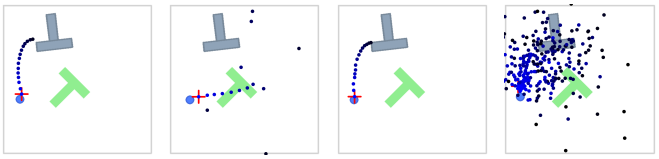
These findings highlight that SFP is particularly well suited to complex, temporally extended tasks, offering both improved task success rates and higher modeling capacity. The superior results across all task categories, particularly in challenging settings, validate shifted flow as a compelling alternative to conventional action chunking strategies.

Table II compares the performance of SFP with that of three baseline methods (DP, SDP, and FMP) on the Push-T task by evaluating the mean score, success rate, latency, and throughput. Despite the simplicity of Push-T, which involves pushing toy blocks, the task itself is difficult due to non-trivial manipulation where specific parts of the block needs to be pushed to translate the block without rotation. SFP achieved the highest mean score of $96.43^{\pm 0.73}$, outperforming all baselines including FMP ($95.26^{\pm 0.51}$), indicating more accurate and efficient trajectory generation. Although the success rate of SFP ($60.00^{\pm 5.17}$) is slightly lower than that of FMP ($65.60^{\pm 2.06}$), it remains significantly higher than that of DP and SDP by maintaining a strong balance between quality and robustness. SFP demonstrated remarkable sampling efficiency, with a latency of merely 5.22ms substantially faster than all baselines including FMP (77.70ms). In terms of throughput, SFP also leads with 191.66 actions per second, far exceeding the performance of action chunking based models.

These results demonstrate that SFP not only delivers high task accuracy but also achieves exceptional computational efficiency. Its closed-loop design and use of shifted flow effectively mitigate the accumulation of errors while enabling real-time robotic control in complex environments.

C. Trajectory Visualization Analysis

We analyzed the intermediate trajectories predicted by each model to compare the effectiveness of shifted flow and action chunking. Figure 3 illustrates these trajectories across all methods. In general, the trajectories can be grouped into three categories. First, DP and FMP exhibited clean trajectories because they were trained to generate complete action chunks. In contrast, SDP displays a mix of clean and noisy chunks, reflecting the partially noisy nature of its action buffer. Notably, because of the use of action chunking, most SDP trajectories remained clean. SDP takes advantage of the fact that action chunking in DP discards the last few actions of each chunk and instead repurposes them to initialize the next denoising step. SFP further exploits this mechanism by predicting only the first action in a chunk as clean, while deferring the remaining actions to future sampling steps. Because the majority of the trajectories of the SFP were noisy, we sampled 16 additional trajectories to analyze the evolution of noise over time. The later time steps, visualized as darker actions, show greater spread, whereas the earlier time steps, shown as brighter actions, remain closer to a plausible trajectory.



(a) DP (b) SDP (c) FMP (d) SFP

Fig. 3: **Visualization of intermediate trajectories.** Each method exhibits distinct trajectory characteristics. DP and FMP produce clean and complete action chunks. SDP generates mostly clean trajectories with occasional noise due to its partially denoised buffer. Proposed SFP yields highly noisy trajectories by design, predicting only the first action cleanly and using the rest for future sampling. Black circles represent actions in the more distant future, indicating increased dispersion, while blue circles indicate earlier, more plausible actions.

These findings validate shifted flow as a promising alternative to action chunking. Importantly, SFP outperformed FMP, despite the only architectural difference being the use of shifted flow instead of action chunking. In the following section, we analyze whether compounding errors is an issue during sequential predictions in SFP.

D. Compounding Errors

As our model predicts actions in a single step, it can be considered autoregressive. Although the proposed model predicts future actions, compounding errors may become a major issue in the absence of chunking. We evaluated SFP with tasks with longer horizons to examine whether compounding errors became an issue. The planning horizon is relatively short for basic and contact-rich tasks. Therefore, we evaluated SFP for long-horizon tasks of the MimicGen benchmark. Long-horizon tasks consist of many subtasks that must be executed sequentially. Additionally, the rollout steps are much longer, where the maximum number of steps for basic and contact-rich tasks at 300, long-horizon tasks goes up to 800 steps. Based on the results in Table I, SFP generally outperforms the other methods, consistent with the results of the basic and contact-rich tasks. This confirms that although shifted flow is similar to autoregressive methods, SFP does not suffer from compounding errors.

E. Ablation Studies

Window Size Analysis: We conducted an ablation study to assess the effect of sampling window size (i.e., the action horizon) on model performance. As shown in Table III, a window size of 16 offers the best trade-off between capturing meaningful temporal dependencies and avoiding overfitting. Shorter horizons (e.g., 4 or 8) fail to encode sufficient temporal context, while excessively long horizons (e.g., 32) introduce redundant or noisy signals that degrade performance. Notably, this validates a central design choice in our SFP, which benefits from temporal structure without

Window Size	Mean Score	Success Rate
4	$76.10^{\pm 1.96}$	$38.00^{\pm 4.29}$
8	$77.21^{\pm 1.78}$	$33.20^{\pm 5.18}$
16	$94.78^{\pm 1.02}$	$62.60^{\pm 4.11}$
32	$94.52^{\pm 1.21}$	$49.40^{\pm 4.61}$

TABLE III: **Effect of the sampling window size on model performance.** Evaluation of SFP performance across various window size on Push-T. Metrics are computed with 10 rollouts with $\pm$ indicating the 95% confidence interval.A window size of 16 offers the best trade-off between capturing sufficient temporal context and avoiding overfitting, leading to superior performance in terms of both mean score and success rate.

relying on fixed action chunks. Based on these results, we fix the window size to 16 across all experiments to ensure robust and consistent performance.

F. Real World Manipulation Tasks

We reproduced the "three-piece assembly" task from the MimicGen suite on a real robotic platform. This task consists of multiple subtasks and involves the assembly of three blocks in a specific orientation and sequence. For the experiments, we used a serial robotic arm with an R-P-P-P-R-P joint configuration in which the three axes were aligned in parallel.

Fifty demonstration trajectories were collected to train the policy. Although simulation environments often provide ideal conditions such as clean visuals and deterministic physics engines, the real world poses additional challenges owing to sensor noise, mechanical inaccuracies, and environmental noise.

Figure 4 shows the real-world deployment of SFP on the Three-Piece Assembly task from the MimicGen suite. The task involves sequential manipulation of three blocks with precise alignment and ordering. Each subfigure shows a key stage in the assembly process. Despite real-world challenges such as sensor noise and mechanical imperfections, SFP demonstrates smooth and reliable execution without signs of compounding error or instability. The consistent performance highlights the benefit of SFP's closed-loop control and shifted timestep formulation, enabling robust generalization from simulation to the physical environment.

V. CONCLUSION

We introduced shifted flow, a novel flow-based generative model that operates over shifted time steps, offering a new perspective on action sequence modeling for visuomotor control. Building on this formulation, we developed SFP, which generates actions in a manner similar to auto-regressive models yet mitigates the compounding error typically associated with such approaches. By conditioning future time steps and leveraging inherent uncertainty, SFP captures the multimodal nature of expert demonstrations more effectively than prior methods. Experimental results on the MimicGen benchmark confirms that SFP outperforms existing state-of-the-art policies not only in terms of the success rate but also in sampling efficiency. In particular, SFP demonstrates superior generalization to unseen configurations, benefiting from its temporally structured prediction mechanism.

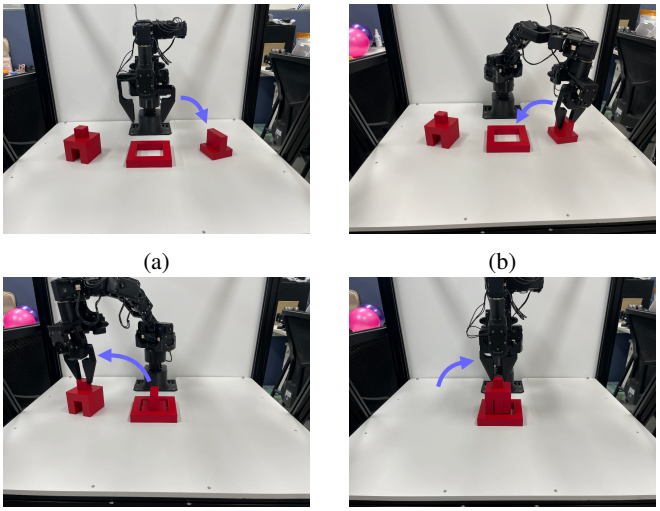


Fig. 4: **Qualitative results of real-world manipulation using SFP on the Three-Piece Assembly task.** The robot completed all subtasks reliably, with no visible compounding error.

Overall, our findings position shifted flow as a compelling alternative to action chunking, capable of serving as a drop-in replacement for flow-based visuomotor policies. Future research could explore the extension of this architecture to integrate language and visual signals, paving the way for more general-purpose robotic agents.

ACKNOWLEDGMENTS

This research was supported by the National Research Foundation of Korea (NRF) grant funded by the Ministry of Education of the Republic of Korea (No. RS-2025-25399600) and Basic Science Research Program through the NRF funded by the Ministry of Education (RS-2024-00410940)

REFERENCES

[1] C. Chi et al., "Diffusion policy: Visuomotor policy learning via action diffusion," *The International Journal of Robotics Research*, doi.org/10.1177/02783649241273668, 2023.

[2] T. Pearce et al., "Imitating human behaviour with diffusion models," in *Deep Reinforcement Learning Workshop NeurIPS 2022*, 2022, pp. 1–24.

[3] M. Reuss, M. Li, X. Jia, and R. Lioutikov, "Goal-conditioned imitation learning using score-based diffusion policies," in *Robotics: Science and Systems*, 2023, pp. 1–13.

[4] S. H. Høeg, Y. Du, and O. Egeland, "Streaming diffusion policy: Fast policy synthesis with variable noise diffusion models," 2025, pp. 1–8.

[5] A. Prasad, K. Lin, J. Wu, L. Zhou, and J. Bohg, "Consistency policy: Accelerated visuomotor policies via consistency distillation," 2024, pp. 1–13.

[6] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," *arXiv preprint arXiv:2304.13705*, 2023.

[7] S. H. Høeg, Y. Du, and O. Egeland, "Streaming diffusion policy: Fast policy synthesis with variable noise diffusion models," *arXiv preprint arXiv:2406.04806*, 2024.

[8] A. Prasad, K. Lin, J. Wu, L. Zhou, and J. Bohg, "Consistency policy: Accelerated visuomotor policies via consistency distillation," in *Robotics: Science and Systems*, 2024.

[9] F. Zhang and M. Gienger, "Affordance-based robot manipulation with flow matching," *arXiv preprint arXiv:2409.01083*, 2024.

[10] Q. Zhang, Z. Liu, H. Fan, G. Liu, B. Zeng, and S. Liu, "Flowpolicy: Enabling fast and robust 3d flow-based policy via consistency flow matching for robot manipulation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, 2025, pp. 14 754–14 762.

[11] A. Mandlekar et al., "What matters in learning from offline human demonstrations for robot manipulation," *arXiv preprint arXiv:2108.03298*, 2021.

[12] P. Florence et al., "Implicit behavioral cloning," in *Conference on robot learning*, PMLR, 2022, pp. 158–168.

[13] A. Van Den Oord, O. Vinyals, et al., "Neural discrete representation learning," *Advances in neural information processing systems*, vol. 30, 2017.

[14] O. M. Team et al., "Octo: An open-source generalist robot policy," *arXiv preprint arXiv:2405.12213*, 2024.

[15] K. Black et al., "π_0: A vision-language-action flow model for general robot control," *arXiv preprint arXiv:2410.24164*, 2024.

[16] A. Mandlekar et al., "Mimicgen: A data generation system for scalable robot learning using human demonstrations," *arXiv preprint arXiv:2310.17596*, 2023.

[17] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," vol. 33, 2020, pp. 6840–6851.

[18] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," 2021, pp. 1–12.

[19] D. Ruhe, J. Heek, T. Salimans, and E. Hoogeboom, "Rolling diffusion models," 2024, pp. 42 818–42 835.

[20] S. Sheynin et al., "Emu edit: Precise image editing via recognition and generation tasks," 2024, pp. 8871–8879.

[21] J. Ho, T. Salimans, A. Gritsenko, W. Chan, M. Norouzi, and D. J. Fleet, "Video diffusion models," vol. 35, 2022, pp. 8633–8646.

[22] Y. Jain, A. Nasery, V. Vineet, and H. Behl, "Peekaboo: Interactive video generation via masked-diffusion," 2024, pp. 8079–8088.

[23] O. Bar-Tal et al., "Lumiere: A space-time diffusion model for video generation," 2024, pp. 1–11.

[24] M. Sun, W. Wang, Z. Qin, J. Sun, S. Chen, and J. Liu, "Glober: Coherent non-autoregressive video generation via global guided video decoder," vol. 36, 2023, pp. 76 120–76 136.

[25] Z. Guo et al., "Audio generation with multiple conditional diffusion model," 16, vol. 38, 2024, pp. 18 153–18 161.

[26] L. Ruan et al., "Mm-diffusion: Learning multi-modal diffusion models for joint audio and video generation," 2023, pp. 10 219–10 228.

[27] S. Wu, J. Wang, W. Ping, W. Nie, and C. Xiao, "Defending against adversarial audio via diffusion model," 2023, pp. 1–21.

[28] Y. Leng et al., "Binauralgrad: A two-stage conditional diffusion probabilistic model for binaural audio synthesis," vol. 35, 2022, pp. 23 689–23 700.

[29] Z. Yan et al., "Dreamdissector: Learning disentangled text-to-3d generation from 2d diffusion priors," 2024, pp. 124–141.

[30] H. Hu, Z. Zhou, V. Jampani, and S. Tulsiani, "Mvdfusion: Single-view 3d via depth-consistent multi-view generation," 2024, pp. 9698–9707.

[31] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," 2022, pp. 10 684–10 695.

[32] C. Fu et al., "A latent diffusion model for protein structure generation," 2024, pp. 1–29.

[33] K. Pnvr, B. Singh, P. Ghosh, B. Siddiquie, and D. Jacobs, "Ld-znet: A latent diffusion approach for text-based image segmentation," 2023, pp. 4157–4168.

[34] D. Kim et al., "Consistency trajectory models: Learning probability flow ode trajectory of diffusion," 2024, pp. 1–33.

[35] Y. Song, P. Dhariwal, M. Chen, and I. Sutskever, "Consistency models," 2023, pp. 32 211–32 252.

[36] S. Lee, Y. Wang, H. Etukuru, H. J. Kim, N. M. M. Shafiullah, and L. Pinto, "Behavior generation with latent actions," *arXiv preprint arXiv:2403.03181*, 2024.

[37] N. M. Shafiullah, Z. Cui, A. A. Altanzaya, and L. Pinto, "Behavior transformers: Cloning k modes with one stone," *Advances in neural information processing systems*, vol. 35, pp. 22 955–22 968, 2022.

[38] B. Zitkovich et al., "Rt-2: Vision-language-action models transfer web knowledge to robotic control," in *Conference on Robot Learning*, PMLR, 2023, pp. 2165–2183.

[39] M. J. Kim et al., "Openvla: An open-source vision-language-action model," *arXiv preprint arXiv:2406.09246*, 2024.

[40] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," 2023.