arXiv:2409.07163v3 [cs.RO] 3 Aug 2026

Mamba Policy: Towards Efficient 3D Diffusion Policy with Hybrid Selective State Models

Jiahang Cao[1,3*], Qiang Zhang[1,3*], Jingkai Sun[1], Jiaxu Wang[1], Hao Cheng[1], Yulin Li[2], Jun Ma[2], Kun Wu[3], Zhiyuan Xu[3], Yecheng Shao[4], Wen Zhao[3], Gang Han[3], Yijie Guo[3], Renjing Xu[1†]

***Abstract*— Diffusion models have been widely employed in the field of 3D manipulation due to their efficient capability to learn distributions, allowing for precise prediction of action trajectories. However, diffusion models typically rely on large parameter UNet backbones as policy networks, which can be challenging to deploy on resource-constrained devices. Recently, the Mamba model has emerged as a promising solution for efficient modeling, offering low computational complexity and strong performance in sequence modeling. In this work, we propose the Mamba Policy, a *lighter but stronger* policy that reduces the parameter count by over 80% compared to the original policy network while achieving superior performance. Specifically, we introduce the XMamba Block, which effectively integrates input information with conditional features and leverages a combination of Mamba and Attention mechanisms for deep feature extraction. Extensive experiments demonstrate that the Mamba Policy excels on the Adroit, Dexart, and MetaWorld datasets, requiring significantly fewer computational resources. Additionally, we highlight the Mamba Policy's enhanced robustness in long-horizon scenarios compared to baseline methods and explore the performance of various Mamba variants within the Mamba Policy framework. Real-world experiments are also conducted to further validate its effectiveness. Our open-source project page can be found at https://sagecao1125.github.io/mamba_policy/.**

I. Introduction

Visuomotor policies, which involve the seamless integration of visual perception with motor control, are crucial for enabling robots to perform complex tasks based on visual inputs. Within this framework, imitation learning has emerged as an effective approach. By observing and mimicking human demonstrations, robots can learn a range of skills, such as dexterous hand control [1], [2], grasping [3], [4], [5], and locomotion [6], [7], [8]. Imitation learning simplifies the transfer of human expertise to robotic systems, making it a powerful tool for developing robots capable of performing intricate tasks.

Among imitation learning methods, diffusion policy [3] has recently attracted considerable attention. Its strength lies in its ability to effectively manage multimodal action distributions, a key factor in producing diverse and adaptable behaviors in complex environments. This makes diffusion policy particularly advantageous for robotic manipulation tasks, where flexibility and generalization are critical. Consequently, diffusion policies have been widely adopted across various robotic applications [9], [10], [11], [12], consistently demonstrating impressive performance in diverse scenarios.

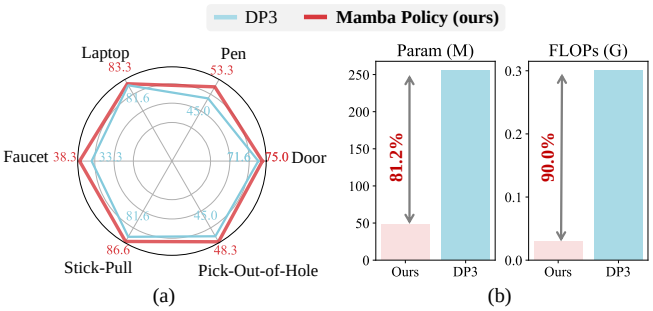


Fig. 1: **Comparison with the SOTA baselines regarding accuracy and computational usage.** Our proposed Mamba Policy (a) achieves superior success rates and (b) offers up to 90% computational savings in terms of floating point operations (FLOPs).

However, despite their success, diffusion policies typically depend on large backbone architectures. For instance, the 3D Diffusion Policy (DP3 [4]) utilizes UNet models with over 200 million parameters. While these architectures excel at capturing intricate details and delivering high performance, their significant computational demands present challenges, especially in resource-constrained environments or when deployed on edge devices. Moreover, the ability to achieve efficient long-horizon prediction is also crucial in robot learning, as it enables more accurate decision-making and planning over extended periods, which is essential for complex tasks such as navigation [13], [14] and manipulation [15], [16]. These considerations highlight the need to develop efficient models that maintain the high performance of diffusion policies while significantly reducing computational overhead and improving long-term prediction capabilities.

One of the recent advancements, Mamba [17], provides valuable insights into addressing these challenges with its selective state space model (SSM), which demonstrates low computational complexity while maintaining robust sequence

†Corresponding author; *Equal contribution.
[1]Jiahang Cao, Qiang Zhang, Jingkai Sun, Jiaxu Wang, Hao Cheng, and Renjing Xu are with the Microelectronics Thrust, The Hong Kong University of Science and Technology (Guangzhou), China. jcao248@connect.hkust-gz.edu.cn; renjingxu@ust.hk
[2] Yulin Li and Jun Ma are with the Division of Emerging Interdisciplinary Areas, The Hong Kong University of Science and Technology, China.
[3]Kun Wu, Zhiyuan Xu, Wen Zhao, Gang Han and Yijie Guo are with the Beijing Innovation Center of Humanoid Robotics, China.
[4]Yecheng Shao is with the Center for X-Mechanics, Zhejiang University, China.

modeling capabilities. This innovation has led to its increasing adoption in various robotics tasks[18], [19], [20], as Mamba excels in trajectory modeling and effectively captures long-horizon dependencies, making it highly effective for complex motion planning and control.

In this work, we introduce Mamba Policy, a **lighter but stronger** policy that reduces the parameter count by over 80% compared to the original policy network while delivering superior performance. We achieve this by integrating a hybrid state space model module with attention mechanisms [21], which we refer to as XMamba. To validate our approach, we conduct extensive experiments across multiple datasets, including Adroit [22], MetaWorld [23], and DexArt [24]. The results demonstrate that Mamba Policy not only significantly outperforms the 3D Diffusion Policy (DP3) in terms of performance but also drastically reduces GPU memory usage. As illustrated in Fig. 1, our method achieves better results with lower computational demands compared to DP3. Additionally, we investigate the impact of horizon length to assess Mamba Policy's capabilities under long-term conditions and explore the effects of various Mamba variants, offering a comprehensive analysis of the effectiveness of our proposed approach.

Our contributions can be summarized as follows:

- We introduce Mamba Policy, a lighter yet stronger policy method based on a hybrid state space model integrated with attention mechanisms.
- Our extensive experiments demonstrate that Mamba Policy achieves up to a $\mathbf{5}\%$ improvement in success rate under a variety of manipulation datasets, while reducing the parameter count by $\mathbf{80}\%$.
- We explore the effects of different horizon lengths, confirming Mamba Policy's stability in long-term scenarios, and analyze the impact of various SSM variants on model performance.

II. Related Work

A. Diffusion Models in Robotic Manipulation

Diffusion models, initially introduced in image generation, have recently gained significant traction in robotic manipulation due to their ability to generate complex and realistic trajectories by denoising random noise into desired actions or paths. Their flexibility and effectiveness have led to widespread adoption in various robotic tasks. Existing approaches can be broadly categorized into two main types based on the sensory information utilized. The first category relies on 2D visual input, where diffusion models generate manipulation actions from RGB images. Chi *et al.* introduced the Diffusion Policy (DP [3]), where, instead of directly outputting an action, the policy infers the action-score gradient conditioned on 2D visual observations. Numerous works have been built on DP, applying it to a range of robotic tasks, such as grasping [11] and navigation [12]. The second category leverages 3D visual information, such as point clouds, which offer a richer and more detailed representation of the environment. Recent 3D-based policies have achieved significant success in control tasks [25], [26], [4], [27]. In this work, our proposed Mamba Policy is based on 3D visual perception.

B. State Space Models

State Space Models (SSMs) represent a modern category of sequence models that draw inspiration from specific dynamic systems. To elucidate the process of modeling, we meticulously describe the architecture of Structured State Space models (S4) as follows:

$$h'(t) = \mathbf{A}h(t) + \mathbf{B}x(t), \tag{1}$$
$$y(t) = \mathbf{C}h(t) + \mathbf{D}x(t), \tag{2}$$

with the quartet $(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$ steering the entire continuous framework. To handle discrete sequences, it's essential to convert S4 into a discretized variant:

$$h'(t) = \overline{\mathbf{A}}h(t) + \overline{\mathbf{B}}x(t), \tag{3}$$
$$y(t) = \mathbf{C}h(t), \tag{4}$$

wherein S4 employs the Zero-Order Hold (ZOH) method for discretization, defined as $\overline{\mathbf{A}} = \exp(\Delta\mathbf{A})$ and $\overline{\mathbf{B}} = (\Delta\mathbf{A})^{-1}(\exp(\Delta\mathbf{A}) - \mathbf{I})(\Delta\mathbf{B})$. Here, $\mathbf{D}$ is conceptualized as parameter-driven skip connections, hence simplified to 0 for brevity. Following the transformation from $(\Delta, \mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$ to $(\widetilde{\mathbf{A}}, \widetilde{\mathbf{B}}, \mathbf{C})$, the model can be executed through two perspectives: (a) a linear recurrence approach requiring only $O(1)$ complexity during inference and (b) a global convolution strategy that enables swift parallel processing during training sessions. This adaptive capability endows the SSM with a significant efficiency edge, especially when juxtaposed against traditional sequence models such as transformer [21], [28], [29] and RWKV [30] in NLP endeavors.

C. Mamba and its Variants

To elevate both selectivity and awareness of context, the introduction of Mamba [17] aims at tackling intricate sequential challenges. Mamba refines the S4 framework by modifying its core parameters to be time-dependent, as opposed to being time-independent. Such change significantly extends the utility of State Space Models (SSMs) to a wide array of fields, including, but not limited to, the realms of vision [31], [32], [33], [34], NLP [35], [36], and healthcare [37], [38].

Many works have focused on improving the Mamba framework. Dao *et al.* [39] developed a comprehensive framework of theoretical connections between SSMs and variants of attention, leading to the design of Mamba2 with state space duality, which achieves a 2-8$\times$ speedup compared to the original Mamba. Vision Mamba [32] introduced bidirectional state space models by incorporating an additional backward branch to compress visual representations. Hydra [40] introduced a quasiseparable matrix mixer to establish a bidirectional extension of Mamba. In this paper, we also experiment with different Mamba variants and provide a detailed comparison of their performance.

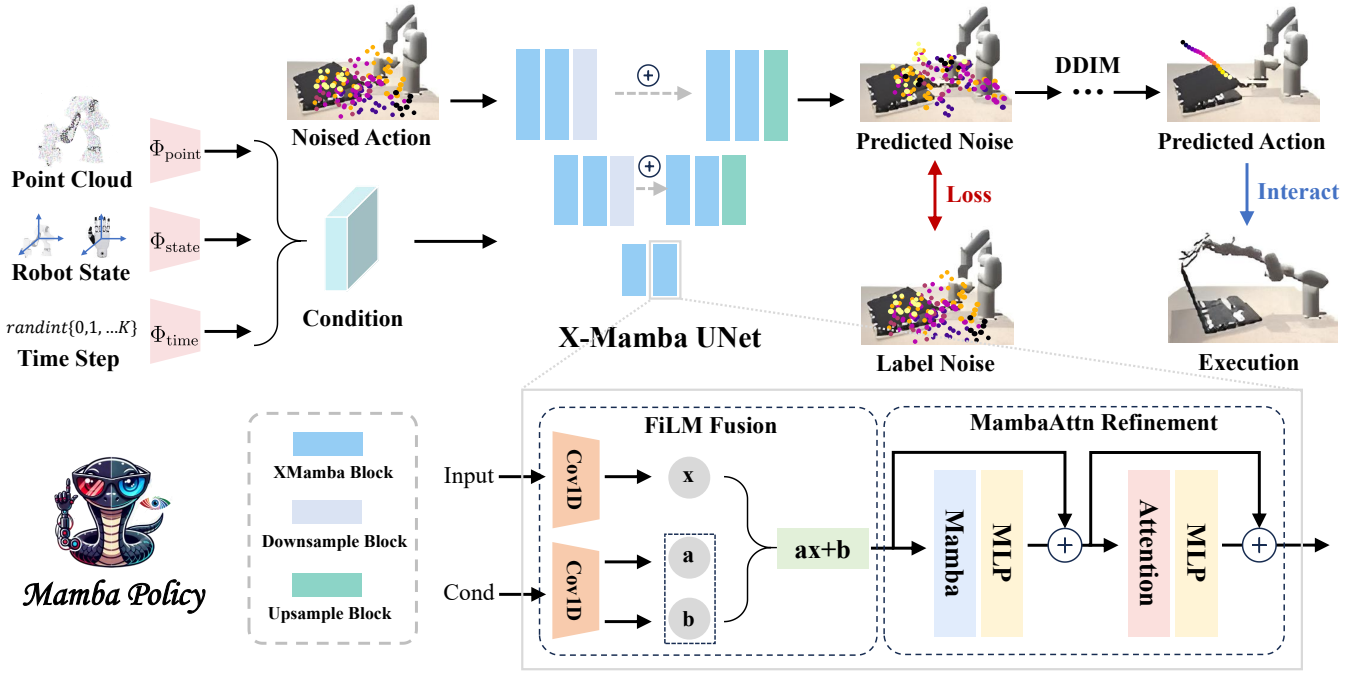


Fig. 2: **Overview of Mamba Policy.** Our proposed model takes the noised action and the condition as inputs, the latter of which is composed of three parts: point cloud perception embedding, robot state embedding, and time embedding. Each of these components is processed through its respective encoder Φ_{type}. The X-Mamba UNet is then employed to process these inputs and ultimately return the predicted noise, with XMamba blocks serving as a key role. During training, the model is updated using MSE loss (Eq. 16) with the label noise. For validation, the model leverages DDIM to reconstruct the original action, which is then used to interact with the environment and execute different tasks.

III. Preliminaries

A. Diffusion Models

Diffusion models are a class of generative models that have been widely applied in various domains. The core idea behind diffusion models is to gradually transform a simple noise distribution into a complex data distribution through a sequence of steps. This is achieved through two main processes: the forward process and the reverse process. Next, we introduce the details of DDPM [41].

Forward Process. The forward process gradually adds noise to the data in a series of T steps. Starting with a data sample $\mathbf{x}_0$, Gaussian noise is incrementally added to the sample at each time step t. The process is defined by a Markov chain:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}), \tag{5}$$

where $\beta_t \in (0,1)$ is a variance schedule that controls the amount of noise added at each step. The forward process can be formulated as: $q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})$.

Reverse Process. The reverse process aims to recover the original data sample $\mathbf{x}_0$ from the noisy sample $\mathbf{x}_T$. This is done by learning a reverse Markov chain parameterized by a neural network:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)), \tag{6}$$

where $\mu_\theta(\mathbf{x}_t, t)$ and $\Sigma_\theta(\mathbf{x}_t, t)$ are the mean and covariance at each step t. The reverse process can be expressed as: $p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$. The training of the diffusion model involves optimizing the parameters θ by minimizing the variational bound on the negative log-likelihood of the data, which typically reduces to a mean-squared error (MSE) loss between the true noise and the predicted noise:

$$\mathcal{L}_{\text{ddpm}} = \left|\left|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right|\right|_2^2, \tag{7}$$

where $\bar{\alpha}_t$ depends on the noise scheduler.

It is important to note that our approach employs DDIM [42], which can be viewed as an extension of DDPM. The detailed operational process and advantages of using DDIM will be discussed in subsequent sections.

IV. Our Method: Mamba Policy

A. Overview

As illustrated in Fig. 2, the Mamba policy is divided into two parts: perception extraction and decision prediction. In the perception stage, we utilize the simple MLP encoder [4] to process single-view point clouds for perception extraction. The extracted features, combined with self-state features and time embeddings, are then fed into the X-Mamba UNet, thereby accomplishing decision prediction. Similar to the Diffusion Policy [3], we introduce the concepts of total prediction horizon T, observation length T_o, and action prediction length T_a. At time step t, the Mamba policy ingests the most recent T_o steps of observation data as its input and forecasts T steps of action, wherein T_a steps of actions starting from t are as the output.

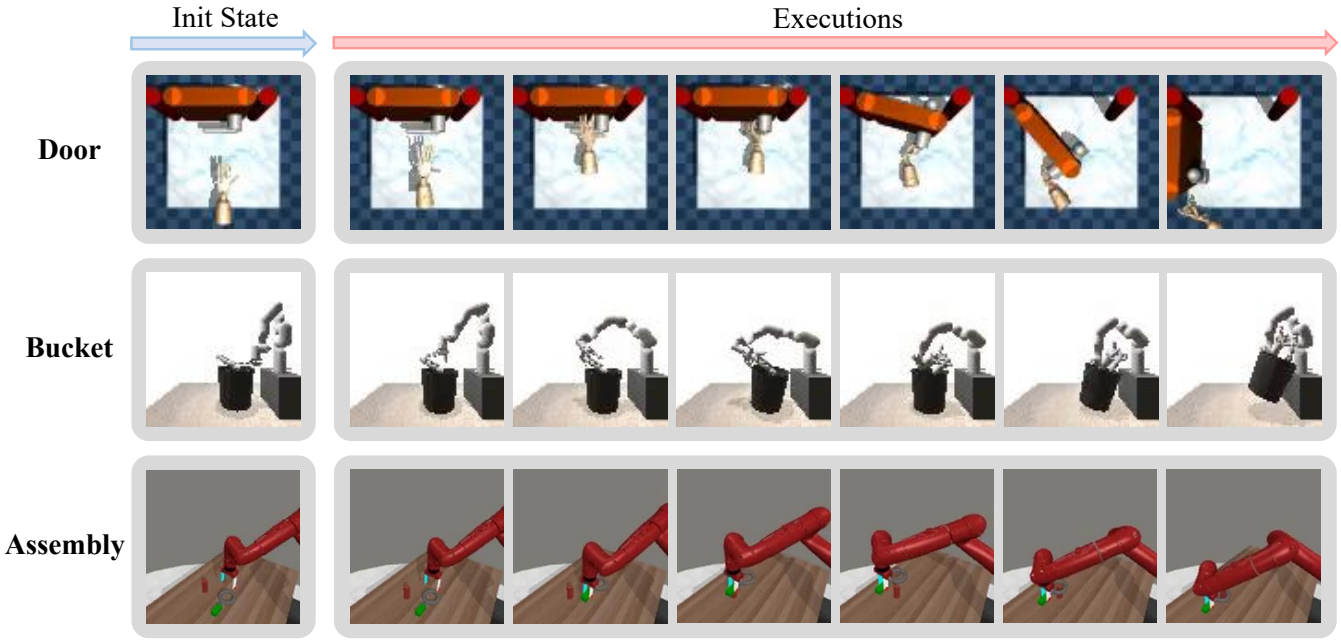


Fig. 3: **Visualization of our manipulation results.** We conduct experiments on three datasets, including Adorit, MetaWorld, and DexArt. Here we illustrate the results in Adroit Door (top), DexArt Bucket (middle), and MetaWorld Assembly (bottom). During the interaction, our proposed Mamba Policy outputs future execution actions until the task is successfully implemented.

B. XMamba

In this study, we develop an innovative denoising network: X-Mamba UNet, with XMamba blocks serving as a key role. We will next delve into the operational process of the XMamba: Suppose the input perception feature $c \in \mathbb{R}^D$ is processed through the DP3 encoder [4] with point cloud and self-state information, the noise input $a \in \mathbb{R}^d$ is randomly initialized from Gaussian distribution, where D and d denote the embedding and action dimension, respectively. c and a will be processed through N XMamba block that consists of the fusion stage and refinement stage, and eventually return the prediction.

FiLM Fusion: To effectively integrate the perception features with the input, we employ the Feature-wise Linear Modulation (FiLM [43]) method, where the specific process is presented as follows:

$$x_{in} = \sigma(\texttt{GN}(\texttt{Conv1D}(a))), \tag{8}$$
$$x_s, x_b = f(\texttt{GN}(\sigma(c))), \tag{9}$$
$$o_1 = x_s * x_{in} + x_b, \tag{10}$$

where σ denotes the mish [44] activation function, $\texttt{GN}$ means the group normalization [45], f denotes the split function which divides the tensor into two parts.

MambaAttn Refinement: Next, we further refine the features using the Mamba and Attention module. We begin by presenting the definition of the mixer module:

$$\mathcal{F}(x; \texttt{Mixer}):$$
$$\dot{y} = x + \texttt{Drop}(\texttt{Mixer}(x) * \gamma_1), \tag{11}$$
$$y = \dot{y} + \texttt{Drop}(\texttt{MLP}(\dot{y}) * \gamma_2), \tag{12}$$

where $\texttt{Drop}$ denotes the drop path [46] strategy, γ_i denotes the trainable parameter to control the feature scale. Then, the fusion feature o_1 is processed through:

$$o_2 = \mathcal{F}(o_1; \texttt{Mamba}), \tag{13}$$
$$o_3 = \mathcal{F}(o_2; \texttt{Attention}), \tag{14}$$

where we adopt the standard Mamba [17] and attention [21] block. After processing through N XMamba blocks, a simple decoder is utilized to map the feature into our desired shape:

$$a_{out} = \texttt{Conv1D}\big(\sigma(\texttt{GN}(\texttt{Conv1D}(o_3)))\big). \tag{15}$$

C. Decision Making

The decision module of our Mamba Policy is based on the conditional diffusion models. In this section, we will introduce the training and inference process of our Mamba Policy. We simplify the aforementioned XMamba UNet as $\boldsymbol{\epsilon}_\theta$, and we use the DDIM [42] as our diffusion solver.

Training: The training process starts by randomly drawing action samples $\mathbf{a}_0$ from raw dataset. In the denoising process, the input will be modified into noised action with a random noise $\boldsymbol{\epsilon} \in \mathcal{N}(0, \mathbf{I})$ for iteration t. The denoise network $\boldsymbol{\epsilon}_\theta$ will then predict the noise with input noised and the perception condition. As described in DDIM [42], the objective is to minimize the KL-divergence between the origin data distribution and the generated sample distribution, we modify the loss function by adding the condition c as follows:

$$\mathcal{L}_{\text{ddim}} = \left|\left|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{a}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, c, t)\right|\right|_2^2, \tag{16}$$

where $\bar{\alpha}_t$ depends on the noise scheduler.

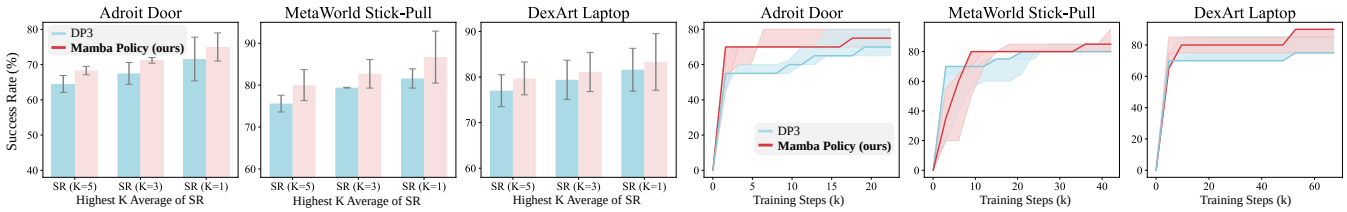


Fig. 4: **Visualization of Success Rates and Training Curves.** We visualize the comparisons in terms of different highest K average of success rates, where our proposed Mamba Policy achieves superior results. The stable training curves also demonstrate the effectiveness of our model.

TABLE I: **Quantitative comparisons of different baselines in simulation environments.** We compare our Mamba Policy with IBC, BCRNN, 3D Diffusion Policy, and Diffusion Policy in Adroit, MetaWorld and DexArt datasets, in terms of SR_5. † denotes our reproduced results for fair comparison. Our proposed Mamba Policy achieves superior results across all domains.

	Adroit			**MetaWorld**			**DexArt**				
Method	Hammer	Door	Pen	Assembly	Disassemble	Stick-Push	Laptop	Faucet	Toilet	Bucket	**Average**
Diffusion Policy [3]	48±17	50±5	25±4	15±1	43±7	63±3	69±4	23±8	58±2	46±1	44.0
BCRNN [47]	8±14	0±0	8±1	1±5	11±6	0±0	29±12	26±2	38±10	24±11	14.5
IBC [48]	0±0	0±0	10±1	18±9	3±5	50±6	1±1	7±2	15±1	0±0	10.4
DP3 [4]†	100.0±0.0	64.5±2.4	41.0±3.3	99.6±0.4	**76.0±2.4**	**100.0±0.0**	77.0±3.5	30.0±2.9	74.3±2.6	27.0±0.8	68.9
Mamba Policy (ours)	**100.0±0**	**68.3±1.2**	**41.0±2.4**	**100.0±0.0**	**76.0±3.7**	**100.0±0.0**	**79.6±3.6**	**33.0±2.1**	**76.3±0.4**	**27.0±0.8**	**70.1**

Inference: After training the denoising network ϵ_θ, we can now approximate the distribution $p_\theta(\mathbf{a}_{t-1}|\mathbf{a}_t)$ using a non-stochastic approach and iteratively denoise the noised action $\mathbf{a}_K$ into the predicted action a_0, where K denotes the diffusion steps. Each iterative step is illustrated through:

$$\mathbf{a}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\left(\frac{\mathbf{a}_t - \sqrt{1-\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{a}_t, c, t)\right) + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\cdot\boldsymbol{\epsilon}_\theta(\mathbf{a}_t, c, t) + \sigma_t\boldsymbol{\epsilon}_t, \tag{17}$$

where $\bar{\alpha}_t, \sigma_t$ are related to the noise scheduler settings, and $\boldsymbol{\epsilon}_t \in \mathcal{N}(0, \mathbf{I})$. With K steps, we can obtain the final action prediction $\mathbf{a}_0$ to interact in the RL environments.

V. Experiment

Datasets. We conduct our experiments in a large variety of datasets, including 3 domains in Adroit [22], 2 domains in MetaWorld [23] with very hard levels, and 4 domains in DexArt [24] environments. We adopt the same data collection method in DP3 [4], where we only generate successful trajectories as expert data.

Baselines. We choose the 3D diffusion policy (DP3 [4]), 2D diffusion policy (DP [3]), BCRNN [47], and IBC [48] as our baselines, where the results are from DP3's origin paper. Since the generation of expert data is subject to randomness, we reproduced DP3 results with marker † by re-generated dataset to ensure fair comparisons.

Experiment Settings. In the primary experiments, we set the prediction horizon to 4, the observation length to 2, and the action prediction length to 3. The X-Mamba UNet is configured with dimensions [128, 256, 512]. Training parameters are consistent with those used in DP3, including a DDIM noise scheduler with a total timestep of 100 and an inference timestep of 10. The AdamW optimizer is utilized with an initial learning rate of 1e-4, and a cosine learning rate scheduler is employed. The model is trained for a total of 3000 epochs with a batch size of 128.

Evaluation Metrics. We compute the average of the highest 1, 3, 5 success rates and log them as SR_1, SR_3, SR_5, respectively. For each domain, we run 3 seeds (0, 1, 2) and report the mean and standard deviation of the results across three seeds.

A. Comparisons with the State-of-the-Arts

As shown in Tab. I, we conduct extensive evaluations across a broad spectrum of manipulation datasets to thoroughly assess the effectiveness of our Mamba Policy. The results demonstrate that Mamba Policy significantly surpasses the baseline models, particularly in terms of the SR_5 metric. For instance, in the Adroit Door domain, our model achieves a notable performance of 68.3 compared to 64.5 by DP3, underscoring its superior capabilities. Beyond individual cases, Mamba Policy also consistently exhibits a higher overall average performance across all evaluated domains. To provide a more detailed understanding of the model's strengths, Fig. 4 presents a comparative analysis of the SR_1, SR_3, and SR_5 metrics. These metrics are indicative of the model's performance under varying levels of challenge, where a smaller K in SR_K reflects the upper bound of the model's capabilities, and a larger K highlights its average performance across tasks. Notably, Mamba Policy excels across all metrics, showcasing its robustness and adaptability. Additionally, we have visualized the training process to illustrate the stability and reliability of the model's training, further reinforcing the effectiveness of our approach.

B. Efficiency Analysis

To assess the computational efficiency of our proposed model, we conduct an efficiency analysis based on the results shown in Tab. II. The analysis reveals that our model not only achieves better performance but does so with significantly

TABLE II: **Efficiency comparisons on Mamba Policy and 3D Diffusion Policy.** We conduct experiments in difficult MetaWorld environments, where our method achieves stronger results with **80**% fewer parameters and **127**% smaller GPU usage. We abbreviate `very hard` and `hard` levels as `VH` and `H`.

Method	# Param (M)	FLOPs (G)	Torch Allocated Memory (MB)	Total GPU Usage (MB)	Hard Environments in MetaWorld		
					Stick-Pull (VH)	Disassemble (VH)	Pick-Out-Of-Hole (H)
DP3	255.1	0.30	996.1	8306	81.6	80.0	45.0
Mamba Policy	47.9 (↓ **81.2**%)	0.03 (↓ **90.0**%)	137.7 (↓ **86.2**%)	3648 (↓ **127.7**%)	**86.7** (↑ **5.1**%)	**81.7** (↑ **1.7**%)	**48.3** (↑ **3.3**%)

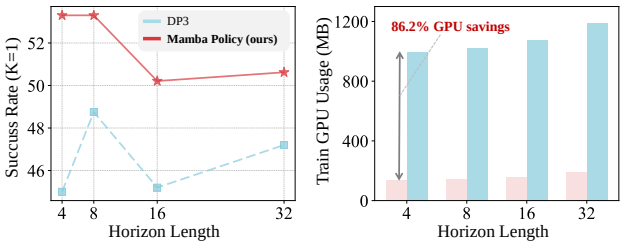


Fig. 5: **Ablation study on different horizon length.** To validate the ability to process longer historical dependencies, we conduct experiments with various horizon lengths and our Mamba Policy achieved robust accuracy and reduced GPU usage compared with DP3, demonstrating the effectiveness and efficiency of our method under long-term scenarios.

TABLE III: **Ablation study on Mamba Policy with different SSM variants.** We include Mamba, Mamba2, bidirectional SSM, and Hydra for comparisons, where the V1-based and Hydra-based policy achieves good performances.

Method	Adroit-Pen		
	SR_5	SR_3	SR_1
DP3	41.0±3.3	42.7±4.0	45.0±4.0
Mamba Policy + Mamba-V1 [17]	41.0±2.4	**45.3**±1.9	**53.3**±2.3
Mamba Policy + Mamba-V2 [39]	41.5±4.5	43.1±4.7	45.0±4.0
Mamba Policy + Bi-SSM[32]	38.9±4.0	43.0±5.7	48.3±8.4
Mamba Policy + Hydra [40]	**41.8**±6.4	45.2±6.5	**53.3**±6.2

lower computational demands. For example, under the Adorit Pen environment, the model parameters are reduced by 81.2%, from 255.1 to 47.9 M, and memory usage decreased by 90.0% regarding the number of floating point operations (FLOPs). Additionally, we record the PyTorch allocated GPU memory per loss backward during training and the required GPU usage dropped by 86.2%. The total GPU usage with our Mamba Policy is 127.7% smaller that the DP3's. Despite these reductions, our model improves the success rate (K=1) by 5.1% in Stick-Pull, 1.7% in Disassemble, and 3.3% in Pick-Out-Of-Hole. These results underscore the efficiency of our model, demonstrating its ability to deliver superior performance with much lower computational costs.

C. Ablation Study

Ablations on SSM Variants. As detailed in Tab. III, we conduct an ablation study by integrating different structured SSM variants into our model. Incorporating Mamba-V1 results in notable improvements, especially in the SR_3 and SR_1 metrics. The Mamba-V2 policy achieves a marginal improvement compared to DP3. In addition, using bidirectional SSM negatively impacts performance. In contrast, Hydra policy delivers superior results compared to the baselines, securing top-1 SR_5 and SR_1 outcomes. However, empirical results indicate that Hydra is very slow to train. In conclusion, Hydra performs best when considering only the end result, but Mamba-V1 offers better overall performance when the time factor is taken into account.

TABLE IV: **Ablation study on key components on Adorit Door environment.** The results underscore the necessity of each proposed component.

Baseline	Mamba	Attention	MLP	SR_5	SR_3	SR_1
✓				61.3±2.6	65.0±2.9	71.6±6.2
	✓		✓	66.9±1.4	69.1±1.7	71.6±2.3
		✓	✓	54.8±6.1	56.9±7.2	61.6±8.4
	✓	✓		61.0±0.8	64.4±2.1	68.3±2.3
	✓	✓	✓	**68.3**±1.2	**71.2**±0.8	**75.0**±4.0

Ablations on Horizon Lengths. In our ablation study on different horizon lengths, we compare the performance of our model against a baseline across various sequence lengths: 4, 8, 16, and 32. Our model consistently outperforms the baseline at each length, demonstrating its robustness across varying temporal horizons. Specifically, for a length of 4, our model achieves a performance score of 53.3, significantly higher than the baseline's 45.0. These results suggest that our model's architecture is better suited for handling longer temporal dependencies, yielding more accurate predictions and enhanced robustness in long-horizon scenarios.

Ablations on Key Components. We conduct a comprehensive ablation study to evaluate the contributions of the core components in our proposed model, where the baseline only contains the FiLM fusion module. As illustrated in Tab. IV, adding the Mamba module alone shows a significant improvement, validating its role in enhancing model robustness. However, with Attention alone or MambaAttn without MLP gains accuracy decreases. The full model including all components achieves the highest success rates, indicating the critical importance of adaptive attention in focusing on relevant features and underscoring the necessity of each proposed method in this work.

VI. Real World Experiment

Experiment Settings. In the real-world experiment, we utilize a Franka arm equipped with a gripper, along with a Realsense D455 camera to capture observations during the task. Inference is performed using a 4090 GPU. Our designed task involved the robot grasping a cup and placing it accurately on a cup holder. The robot is required to execute the task with high precision, both in terms of grasping the cup and positioning it correctly on the holder. Data collection

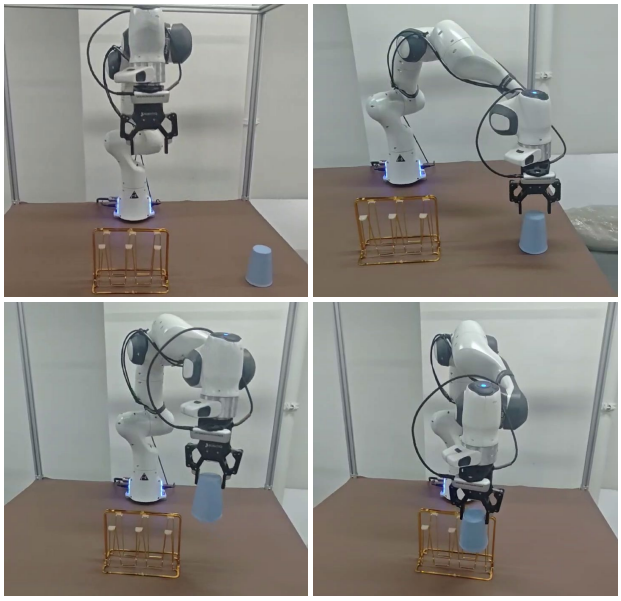


Fig. 6: **Real-world experiment.** The robot successfully grasps the cup and places it on the cup holder as part of the 3D manipulation task. The sequence of images demonstrates the robot's precise grasping and placement actions, validating the effectiveness of the proposed Mamba Policy in real-world robotic manipulation tasks.

is carried out via human teleoperation, where a total of 40 demonstrations are gathered for training the model.

Results. As illustrated in Fig. 6, our Mamba policy demonstrated a high level of precision in executing the grasping operation and successfully placing the cup on the holder. The results validate the effectiveness and practical applicability of the proposed method, showcasing the robot's ability to perform the task with accuracy and reliability in a real-world environment.

VII. Conclusion

In this paper, we introduced the Mamba Policy, a lightweight yet efficient model tailored for 3D manipulation tasks. By significantly reducing the parameter count by over 80% compared to 3D diffusion policy, the Mamba Policy maintains strong performance while being more suitable for deployment on resource-constrained devices. Central to this approach is the XMamba Block, which effectively combines input data with conditional features through the integration of Mamba and Attention mechanisms. Our extensive experiments on the Adroit, Dexart, and MetaWorld datasets demonstrate the Mamba Policy's superior performance and reduced computational requirements. Additionally, we showed that the Mamba Policy exhibits enhanced robustness in long-horizon scenarios, making it a compelling choice for various real-world applications. This work sets the stage for future research in developing efficient models for 3D manipulation, particularly in environments with limited computational resources. Real-world experiments are also conducted to validate its effectiveness further. Our open-source project page can be found at this link.

References

[1] S. P. Arunachalam, S. Silwal, B. Evans, and L. Pinto, "Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation," in *ICRA*. IEEE, 2023, pp. 5954–5961.

[2] D. Wei and H. Xu, "A wearable robotic hand for hand-over-hand imitation learning," in *ICRA*. IEEE, 2024, pp. 18 113–18 119.

[3] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *RSS*, 2023.

[4] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu, "3d diffusion policy," *RSS*, 2024.

[5] K. Yamane, Y. Saigusa, S. Sakaino, and T. Tsuji, "Soft and rigid object grasping with cross-structure hand using bilateral control-based imitation learning," *IEEE RA-L*, 2023.

[6] X. Cheng, K. Shi, A. Agarwal, and D. Pathak, "Extreme parkour with legged robots," in *ICRA*. IEEE, 2024, pp. 11 443–11 450.

[7] A. Tang, T. Hiraoka, N. Hiraoka, F. Shi, K. Kawaharazuka, K. Kojima, K. Okada, and M. Inaba, "Humanmimic: Learning natural locomotion and transitions for humanoid robot via wasserstein adversarial imitation," in *ICRA*. IEEE, 2024, pp. 13 107–13 114.

[8] M. Seo, S. Han, K. Sim, S. H. Bang, C. Gonzalez, L. Sentis, and Y. Zhu, "Deep imitation learning for humanoid loco-manipulation through human teleoperation," in *Humanoids*. IEEE, 2023, pp. 1–8.

[9] H. Ha, P. Florence, and S. Song, "Scaling up and distilling down: Language-guided robot skill acquisition," in *CoRL*. PMLR, 2023, pp. 3766–3777.

[10] I. Kapelyukh, V. Vosylius, and E. Johns, "Dall-e-bot: Introducing web-scale diffusion models to robotics," *IEEE RA-L*, vol. 8, no. 7, pp. 3956–3963, 2023.

[11] U. A. Mishra, S. Xue, Y. Chen, and D. Xu, "Generative skill chaining: Long-horizon skill planning with diffusion models," in *CoRL*. PMLR, 2023, pp. 2905–2925.

[12] A. Sridhar, D. Shah, C. Glossop, and S. Levine, "Nomad: Goal masked diffusion policies for navigation and exploration," in *ICRA*. IEEE, 2024, pp. 63–70.

[13] I. Navarro, J. Patrikar, J. P. Dantas, R. Baijal, I. Higgins, S. Scherer, and J. Oh, "Sorts: Learned tree search for long horizon social robot navigation," *IEEE RA-L*, 2024.

[14] C. H. Song, J. Kil, T.-Y. Pan, B. M. Sadler, W.-L. Chao, and Y. Su, "One step at a time: Long-horizon vision-and-language navigation with milestones," in *CVPR*, 2022, pp. 15 482–15 491.

[15] Z. Zhang, Y. Li, O. Bastani, A. Gupta, D. Jayaraman, Y. J. Ma, and L. Weihs, "Universal visual decomposer: Long-horizon manipulation made easy," in *ICRA*. IEEE, 2024, pp. 6973–6980.

[16] Y. Lee, E. S. Hu, and J. J. Lim, "Ikea furniture assembly environment for long-horizon complex manipulation tasks," in *ICRA*. IEEE, 2021, pp. 6343–6349.

[17] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," *arXiv preprint arXiv:2312.00752*, 2023.

[18] J. Liu, M. Liu, Z. Wang, L. Lee, K. Zhou, P. An, S. Yang, R. Zhang, Y. Guo, and S. Zhang, "Robomamba: Multimodal state space model for efficient robot reasoning and manipulation," *arXiv preprint arXiv:2406.04339*, 2024.

[19] S. M. Mustafa, O. Rizvi, Z. A. Usmani, and A. B. Memon, "Context-aware mamba-based reinforcement learning for social robot navigation," *arXiv preprint arXiv:2408.02661*, 2024.

[20] J. Cao, Q. Zhang, Z. Wang, J. Wang, H. Cheng, Y. Shao, W. Zhao, G. Han, Y. Guo, and R. Xu, "Mamba as decision maker: Exploring multi-scale sequence modeling in offline reinforcement learning," *arXiv preprint arXiv:2406.02013*, 2024.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *NeurIPS*, vol. 30, 2017.

[22] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," *RSS*, 2017.

[23] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," in *CoRL*. PMLR, 2020, pp. 1094–1100.

[24] C. Bao, H. Xu, Y. Qin, and X. Wang, "Dexart: Benchmarking generalizable dexterous manipulation with articulated objects," in *CVPR*, 2023, pp. 21 190–21 200.

[25] T. Gervet, Z. Xian, N. Gkanatsios, and K. Fragkiadaki, "Act3d: Infinite resolution action detection transformer for robotic manipulation," *CoRL*, 2023.

[26] T.-W. Ke, N. Gkanatsios, and K. Fragkiadaki, "3d diffuser actor: Policy diffusion with 3d scene representations," *arXiv preprint arXiv:2402.10885*, 2024.

[27] K. Rana, J. Haviland, S. Garg, J. Abou-Chakra, I. Reid, and N. Suenderhauf, "Sayplan: Grounding large language models using 3d scene graphs for scalable robot task planning," in *CoRL*, 2023.

[28] T. Dao, D. Fu, S. Ermon, A. Rudra, and C. Ré, "Flashattention: Fast and memory-efficient exact attention with io-awareness," *NeurIPS*, vol. 35, pp. 16 344–16 359, 2022.

[29] T. Dao, "Flashattention-2: Faster attention with better parallelism and work partitioning," *ICLR*, 2023.

[30] B. Peng, E. Alcaide, Q. Anthony, A. Albalak, S. Arcadinho, S. Biderman, H. Cao, X. Cheng, M. Chung, L. Derczynski *et al.*, "Rwkv: Reinventing rnns for the transformer era," in *EMNLP Findings*, 2023, pp. 14 048–14 077.

[31] A. Hatamizadeh and J. Kautz, "Mambavision: A hybrid mamba-transformer vision backbone," *arXiv preprint arXiv:2407.08083*, 2024.

[32] L. Zhu, B. Liao, Q. Zhang, X. Wang, W. Liu, and X. Wang, "Vision mamba: Efficient visual representation learning with bidirectional state space model," *ICML*, 2024.

[33] Y. Liu, Y. Tian, Y. Zhao, H. Yu, L. Xie, Y. Wang, Q. Ye, and Y. Liu, "Vmamba: Visual state space model," *arXiv preprint arXiv:2401.10166*, 2024.

[34] Y. Teng, Y. Wu, H. Shi, X. Ning, G. Dai, Y. Wang, Z. Li, and X. Liu, "Dim: Diffusion mamba for efficient high-resolution image synthesis," *arXiv preprint arXiv:2405.14224*, 2024.

[35] M. Pióro, K. Ciebiera, K. Król, J. Ludziejewski, and S. Jaszczur, "Moe-mamba: Efficient selective state space models with mixture of experts," *ICLR Workshops*, 2024.

[36] Q. Anthony, Y. Tokpanov, P. Glorioso, and B. Millidge, "Blackmamba: Mixture of experts for state-space models," *arXiv preprint arXiv:2402.01771*, 2024.

[37] J. Liu, H. Yang, H.-Y. Zhou, Y. Xi, L. Yu, Y. Yu, Y. Liang, G. Shi, S. Zhang, H. Zheng *et al.*, "Swin-umamba: Mamba-based unet with imagenet-based pretraining," *arXiv preprint arXiv:2402.03302*, 2024.

[38] Z. Xing, T. Ye, Y. Yang, G. Liu, and L. Zhu, "Segmamba: Long-range sequential modeling mamba for 3d medical image segmentation," *arXiv preprint arXiv:2401.13560*, 2024.

[39] T. Dao and A. Gu, "Transformers are ssms: Generalized models and efficient algorithms through structured state space duality," *ICML*, 2024.

[40] S. Hwang, A. Lahoti, T. Dao, and A. Gu, "Hydra: Bidirectional state space models through generalized matrix mixers," *arXiv preprint arXiv:2407.09941*, 2024.

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

[42] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," *ICLR*, 2020.

[43] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer," in *AAAI*, vol. 32, no. 1, 2018.

[44] D. Misra, "Mish: A self regularized non-monotonic activation function," *arXiv preprint arXiv:1908.08681*, 2019.

[45] Y. Wu and K. He, "Group normalization," in *ECCV*, 2018, pp. 3–19.

[46] G. Larsson, M. Maire, and G. Shakhnarovich, "Fractalnet: Ultra-deep neural networks without residuals," *ICLR*, 2016.

[47] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, "What matters in learning from offline human demonstrations for robot manipulation," *CoRL*, 2021.

[48] P. Florence, C. Lynch, A. Zeng, O. A. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson, "Implicit behavioral cloning," in *CoRL*. PMLR, 2022, pp. 158–168.