# CDP: Towards Robust Autoregressive Visuomotor Policy Learning via Causal Diffusion

**Jiahua Ma***
Sun Yat-sen University
majiahua99@gmail.com

**Yiran Qin***
CUHK(SZ); Oxford
yiranqin@link.cuhk.edu.cn

**Yixiong Li**
Sun Yat-sen University
liyx357@mail2.sysu.edu.cn

**Xuanqi Liao**
Sun Yat-sen University
liaoxq7@mail2.sysu.edu.cn

**Yulan Guo**
Sun Yat-sen University
guoyulan@sysu.edu.cn

**Ruimao Zhang**†
Sun Yat-sen University
zhangrm27@mail.sysu.edu.cn

**Abstract:** Diffusion Policy (*DP*) enables robots to learn complex behaviors by imitating expert demonstrations through action diffusion. However, in practical applications, hardware limitations often degrade data quality, while real-time constraints restrict model inference to instantaneous state and scene observations. These limitations seriously reduce the efficacy of learning from expert demonstrations, resulting in failures in object localization, grasp planning, and long-horizon task execution. To address these challenges, we propose Causal Diffusion Policy (*CDP*), a novel transformer-based diffusion model that enhances action prediction by conditioning on historical action sequences, thereby enabling more coherent and context-aware visuomotor policy learning. To further mitigate the computational cost associated with autoregressive inference, a caching mechanism is also introduced to store attention key-value pairs from previous timesteps, substantially reducing redundant computations during execution. Extensive experiments in both simulated and real-world environments, spanning diverse 2D and 3D manipulation tasks, demonstrate that *CDP* uniquely leverages historical action sequences to achieve significantly higher accuracy than existing methods. Moreover, even when faced with degraded input observation quality, *CDP* maintains remarkable precision by reasoning through temporal continuity, which highlights its practical robustness for robotic control under realistic, imperfect conditions. Project page: https://gaavama.github.io/CDP/

**Keywords:** Robotic Manipulation, Autoregressive, Causal Diffusion Policy

## 1 Introduction

Robotic manipulation tasks—such as grasping, placing, and assembling objects—are notoriously difficult to program explicitly due to their high complexity and variability. Recently, imitation learning (*IL*) offers a promising alternative, allowing robots to acquire manipulation skills by mimicking expert demonstrations [1, 2, 3, 4, 5, 6]. By learning from example trajectories, *IL* eliminates the need for manual behavior design, making it particularly effective in contact-rich or dynamic environments where reward functions or controllers are difficult to define.

*Equal contribution. †Corresponding author.

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea.

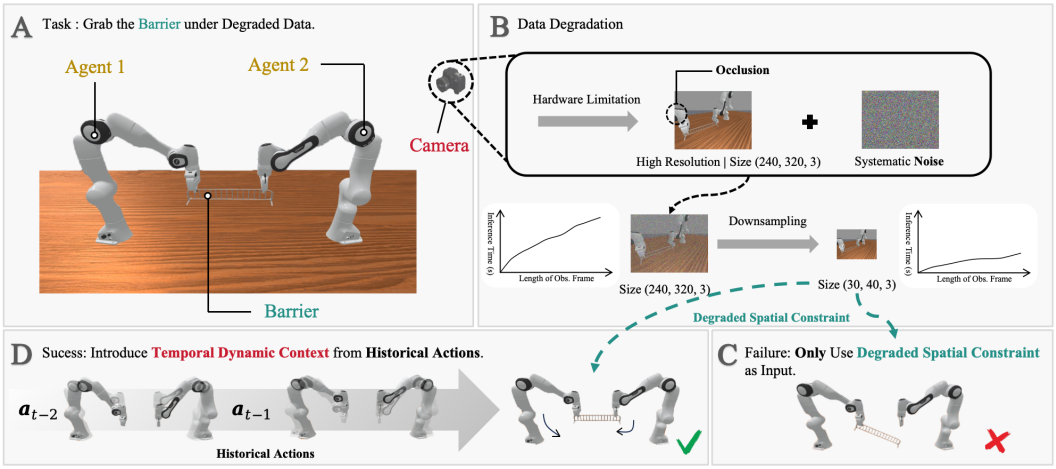



Figure 1: **Causal Diffusion Policy**: a transformer-based diffusion model that enhances action prediction by conditioning on historical action sequences. **A**: When performing the task of "grabbing the barrier" in practice, **B**: the quality of observations is degraded by factors such as sensor noise, occlusions, and hardware limitations. In fact, this degraded but high-dimensional observation data not only fails to provide sufficient spatial constraint information for policy planning but also slows down the planning speed. **C**: In this case, the robot is unable to perform accurate manipulation. **D**: In this paper, we address historical action sequences to introduce temporally rich context as a supplement, which enables more robust policy generation.

As a powerful *IL* framework, Diffusion Policy (*DP*) [7] treats action generation as a denoising diffusion process, which enable smooth and multimodal trajectory prediction, paving the way for more versatile and efficient robot learning system. However, *DP* employs a naive behavior cloning approach to learn the specified tasks, which models actions independently and fails to account for the sequential structure of decision making. This often leads to *distributional shift*, where small prediction errors accumulate and push the robot into unseen or unstable states.

Furthermore, robot actions are continuous and temporally correlated—properties that are difficult to capture under observations common in physical deployments. And in real-world scenarios shown in Fig. 1, sensor noise, occlusions, and hardware limitations degrade observation quality, while real-time inference constraints restrict access to temporally rich context. As a result, *DP* often fails at critical subtasks such as object localization, grasp planning, and long-horizon task execution.

To address these issues, we propose **Causal Diffusion Policy (*CDP*)**, a novel transformer-based diffusion framework that explicitly conditions action prediction on historical action sequences. Rather than relying solely on spatial constraints in instantaneous observations, *CDP* incorporates temporal continuity through an autoregressive causal transformer, enabling the policy to reason over prior actions and their evolving contexts. This design improves coherence, stability, and robustness, especially in challenging conditions where single-frame observations may be unreliable.

To further improve computational efficiency during inference, we introduce a **caching mechanism** that stores and reuses attention key-value pairs computed in earlier timesteps. This avoids redundant computations inherent in standard transformer-based policies and makes the model practical for real-time deployment on physical hardware. Together, causal modeling and inference caching allow *CDP* to generate temporally consistent and high-quality actions with reduced latency. Our contributions can be summarized as follows:

1. We propose *CDP*, a novel transformer-based diffusion framework for robotic manipulation that conditions action prediction on historical action sequences, enabling context-aware, temporally coherent visuomotor policy learning.

2. To enable efficient real-time inference, we introduce a cache sharing mechanism that stores and reuses attention key-value pairs from prior timesteps, substantially reducing computational overhead in autoregressive action generation.
3. Through extensive evaluations on diverse 2D and 3D manipulation tasks in both simulation and real-world settings, we show that *CDP* achieves superior accuracy and robustness over existing methods, especially under degraded observation conditions.

## 2 Related Work

### 2.1 Diffusion Model in Robotic Manipulation

Diffusion Model [8, 9] provides an effective means for robots to acquire human-like skills by emulating expert demonstrations [10, 11, 12, 13, 14, 15, 16]. Recent advancements [17, 18, 19, 20, 21, 22, 23, 24, 25, 26] in this domain have introduced innovative approaches to model visuomotor policies. Diffusion Policy [7] models a visuomotor policy as a conditional denoising diffusion process, i.e. utilizing the robot's observations as the condition to refine noisy trajectories into coherent action sequences. Based on this, 3D Diffusion Policy [27] incorporates simple point-cloud representations, thereby enriching the robot's perception of the environment. Despite these advancements, these holistic approaches that generate complete action sequences in a single pass face inherent limitations, such as potential error propagation and challenges in handling long-range dependencies within the action sequences. To address these issues, recent research has increasingly focused on token-wise incremental generation for robot policy learning. This paradigm shift aims to improve the flexibility and robustness of action generation by breaking down the process into smaller, manageable steps. Notable examples include ICRT [28], ARP [29], and CARP [30]. In this work, we build upon these foundational advancements by proposing a novel transformer-based causal generation model. By incorporating temporal context, our model is better equipped to capture the dynamics of the environment and generate smoother, more coherent action sequences.

### 2.2 Causal Generation in Diffusion Model

Diffusion models have utilized uniform noise levels across all tokens during training. While this approach simplifies the training process, it may compromise the model's ability to capture complex temporal dynamics [31, 32, 33, 34]. To address this limitation, Diffusion Forcing [35] introduced a novel training strategy for sequence diffusion models, where noise levels are independently varied for each frame. This method significantly enhances the model's capacity to generate more realistic and coherent sequences, as demonstrated by both theoretical analysis and empirical results. Building on this foundation, CausVid [36] further extended Diffusion Forcing by integrating it into a causal transformer architecture, resulting in an autoregressive video foundation model. This adaptation effectively combines the strengths of both Diffusion Forcing and transformer-based architectures, thereby achieving improved temporal coherence and consistency in video generation [37, 38, 39, 40, 41, 42, 43, 44]. Ca2-VDM [45] incorporates optimized causal mechanisms and efficient cache management techniques. These innovations reduce computational redundancy by reusing precomputed conditional frames and minimize storage costs through cache sharing across denoising steps, thereby enhancing real-time performance.

## 3 Method

### 3.1 Causal Action Generation

We begin by outlining the training phase of our proposed model, followed by a detailed description of the Causal Action Generation Module (Fig. 4 (a)). Based on this, we introduce the Causal Temporal Attention Mechanism, a pivotal component of the module. This mechanism ensures that each target action can access all historical actions, thereby capturing its temporal dynamics context and enabling the generation of more accurate results. In addition, to enhance the module's robustness

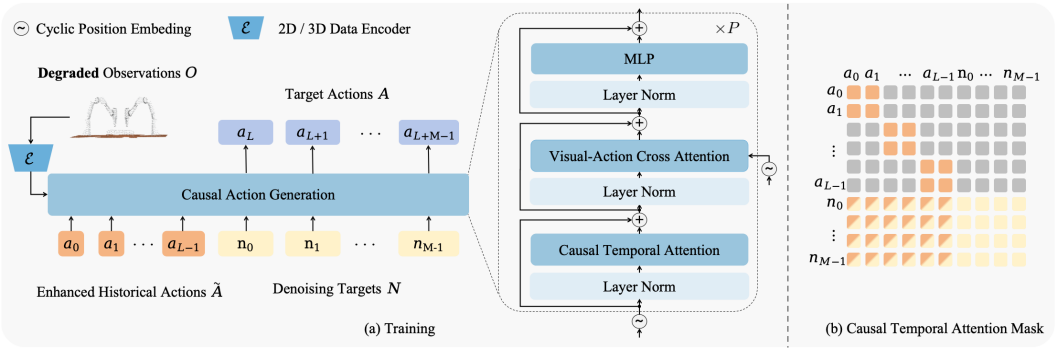



Figure 2: **Causal Action Generation of Our *CDP*.** (a) During training, the Historical Actions $\tilde{\mathbf{A}}$ are combined with the Denoising Targets $\mathbf{N}$. This combined input is then fed into the Causal Action Generation module, which contains $P$ blocks, for denoising. The Target Actions $\mathbf{A}$ are used for training supervision. Before denoising, $\tilde{\mathbf{A}}$ is perturbed by a small-scale noise, which helps to reduce the accumulation of action prediction errors during inference. (b) The Causal Temporal Attention Mask ensures each Denoising Target can access all Historical Actions.

against error-prone historical actions and mitigate the risk of action generation failure due to the accumulation of prediction errors during inference phase, we further integrate the Historical Actions Enhancement. This module significantly improves reliability of the action generation process.

**Training**. During training, the sampled action sequence with the length of $L + M$ is divided into two parts, i.e. the Historical Actions $\tilde{\mathbf{A}} = \{a_k\}_{k=0}^{L-1}$ with the length of $L$, and the Target Actions $\mathbf{A} = \{a_k\}_{k=L}^{L+M-1}$ with the length of $M$. The former serves as the causal condition, which will be perturbed and then, fed into the Causal Action Generation Module together with the Denoising Targets $\mathbf{N} = \{n_k \sim \mathcal{N}(0,1)\}_{k=0}^{M-1}$ and ultimately generate the predicted Target Actions. The objective function for the training of this module is designed to minimize the L2 distance between the predicted Target Actions and its corresponding ground truth,

$$\min \mathbb{E}_{\tilde{\mathbf{A}},\mathbf{A},\mathbf{N}} \|(D_\theta([\tilde{\mathbf{A}}, \mathbf{N}]) - \mathbf{A})\|_2^2 \tag{1}$$

where $[\cdot,\cdot]$ denotes the concatenation operation along the temporal axis and $D_\theta$ denotes the whole denoising process, based on our Causal Action Generation Module with the parameter of $\theta$.

**Historical Actions Re-Denoising**. In autoregressive prediction tasks, errors in historical actions can cause significant deviations that accumulate over time, potentially leading to the failure of robotic manipulation. To address this issue, we perturb the Historical Actions $\tilde{\mathbf{A}}$ by introducing small-scale noise and then, re-denoise them during training.

$$\tilde{\mathbf{A}}^{\texttt{perturb}} = \tilde{\mathbf{A}} + \mathbf{N}_\sigma \tag{2}$$

where $\tilde{\mathbf{A}}^{\texttt{perturb}}$ denotes the perturbed historical actions and $\mathbf{N}_\sigma \sim \mathcal{N}(0, \sigma^2)$ denotes the small-scale noise sequence with the variance of $0 < \sigma < 1$. By introducing $\mathbf{N}_\sigma$, we create a more robust training environment that simulates the occasions where Historical Actions $\tilde{\mathbf{A}}$ may be imperfect or noisy, forcing the model to focus on the coarse-grained temporal dynamic context from the Historical Actions $\tilde{\mathbf{A}}$, instead of depending on their concrete values. These coarse-grained temporal dynamics serve as a complement to the degraded observations $\mathbf{O}$, guiding the subsequent denoising process.

**Causal Action Generation Module**. As depicted in Fig. 4 (a), this module is composed of $P$ blocks. Within each block, we first inject the temporal dynamic context from the Historical Actions $\tilde{\mathbf{A}}$ into the Denoising Targets $\mathbf{N}$ through Causal Temporal Attention (`CTA`), generating intermediate features $\mathbf{N}_t$ . Following this, Visual-Action Cross Attention (`VACA`) is utilized to incorporate spatial constraints from the degraded observation $\mathbf{O}$ into these $\mathbf{N}_t$, generating intermediate features $\mathbf{N}_{ts}$. These $\mathbf{N}_{ts}$ are then further processed by a Multi-layer Perceptron (`MLP`). It is worth noting that we place the denoising timestep embedding $t$ in the `VACA` layer, rather than the `CTA` layer in order

to ensure the cached features of the Historical Actions $\tilde{\mathbf{A}}$ can be correctly shared during the whole denoising process (Sec. 3.2). The entire process within each block can be described as follows,

$$\mathbf{N}_t = \texttt{LN}\left(\texttt{CTA}\left(\tilde{\mathbf{A}},\ \mathbf{N}\right)\right) \tag{3}$$

$$\mathbf{N}_{ts} = \texttt{LN}\left(\texttt{VACA}\left(\texttt{Enc}\left(\mathbf{O}\right),\ \mathbf{N}_t\right)\right) \tag{4}$$

$$\mathbf{N}_o = \texttt{LN}\left(\texttt{MLP}\left(\mathbf{N}_{ts}\right)\right) \tag{5}$$

where $\mathbf{N}_o$ represents the output features of this block. `LN` denotes the Layer Normalization operation, and `Enc` denotes the 2D or 3D data encoder.

**Causal Temporal Attention** (`CTA`). To ensure the causality of action prediction while maintaining compatibility with the inference phase, our Causal Temporal Attention must meet the following criteria: 1) During the autoregressive inference process, new actions are generated based on their historical actions. This means that we only need to ensure that the Denoising Targets $\mathbf{N}$ can attend to the Historical Actions $\tilde{\mathbf{A}}$ during the attention calculation, rather than enforcing causality among the Historical Actions $\tilde{\mathbf{A}}$ themselves. 2) The autoregressive inference process continually discards invalid historical actions, which are too early and too distant from the Target Actions $\mathbf{A}$ temporally. This implies that during the attention calculation, we need to further decouple the Historical Actions $\tilde{\mathbf{A}}$. Specifically, we chunk the Historical Actions $\tilde{\mathbf{A}}$ with a fixed size, ensuring that all actions within a chunk are fully visible to one another, while actions from different chunks cannot attend to each other. 3) Following [7], we increase the redundancy in the length of the Denoising Targets $\mathbf{N}$ to enable the model to anticipate future context while maintaining action coherence throughout the entire inference process. Ultimately, only the first chunk-size actions are utilized as the final predicted Target Actions. To meet the aforementioned constraints, we customize the Causal Temporal Attention Mask, as illustrated in Fig. 4 (b). For the Historical Action $a_0$, the corresponding mask indicates that the attention computation only considers actions within the same chunk, excluding actions from other chunks. Furthermore, all Denoising Targets $n_i$ are treated as being in a single chunk with a larger chunk size than that of the Historical Actions. Their corresponding mask indicates that their attention calculation considers both the Historical Actions and the Denoising Targets themselves.

### 3.2 Chunk-wise Autoregressive Inference with Cache Sharing

In this section, we first provide an overview of our proposed Chunk-wise Autoregressive Inference process (Fig. 3 (a)), which is equipped with a Cache Sharing mechanism (Fig. 3 (b)). Since the entire target action sequence is predicted in an autoregressive manner, its total length remains indeterminate at the outset. To this end, we introduce Cyclic Temporal Embedding to distinguish the temporal order of the whole action sequence (Fig. 3 (b)).

**Chunk-wise Autoregressive Inference.** During the $k$-th autoregressive (`AR-k`) step, we perform denoising using

$$\mathbf{A}^k \sim p_\theta(\mathbf{A}^k|\mathbf{N}^k, \tilde{\mathbf{A}}^k) \tag{6}$$

where $\mathbf{N}^k$, $\tilde{\mathbf{A}}^k$, and $\mathbf{A}^k$ denote the Denoising Targets, Historical Actions, and Target Actions at the `AR-k` step, respectively. Within the causal generation framework, features computation proceeds in a unidirectional manner. Specifically, $\mathbf{N}^k$ is denoised conditioned on $\tilde{\mathbf{A}}^k$, and the cache for $\tilde{\mathbf{A}}^k$ can be precomputed in prior autoregressive steps without regard for $\mathbf{N}^k$. After denoising, the Target Actions $\mathbf{A}^k$ generated in the `AR-k` step are applied to the environment and serve as the partial of Historical Actions in the `AR-k+1` step, as illustrated in Fig. 3 (a). In the `AR-k+1` step, we update the historical actions in a window-sliding manner (Fig. 3 (b)). This involves discarding the invalid part of $\tilde{\mathbf{A}}^k$ and incorporating the Target Actions $\mathbf{A}^k$ to form the Historical Actions $\tilde{\mathbf{A}}^{k+1}$ for the `AR-k+1` step. These Historical Actions $\tilde{\mathbf{A}}^{k+1}$ are then fed into the Causal Action Generation Module to generate the Target Actions $\mathbf{A}^{k+1}$ in the `AR-k+1` step.

**Cache Sharing.** In the `AR-k` step, the Historical Actions $\tilde{\mathbf{A}}^k$ of length $L$ can be divided into two parts: Cached Historical Actions $\tilde{\mathbf{A}}^k_{0:l-1}$ and Uncached Historical Actions $\tilde{\mathbf{A}}^k_{l:L-1}$, based on whether

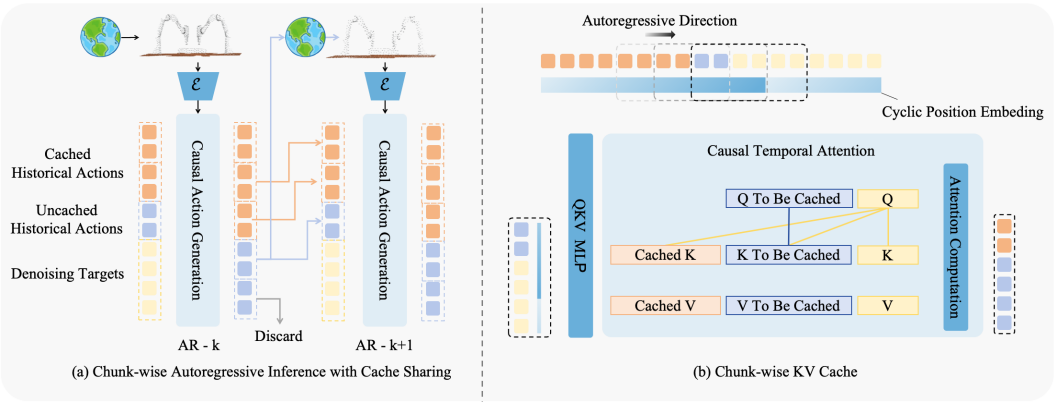



Figure 3: **Chunk-wise Autoregressive Inference of Our *CDP*.** (a) The orange and purple blocks denote actions whose Key and Value representations have been cached and not cached until the current step, respectively. The yellow block denotes the Gaussian noises. During the `AR-k` step, we perform denoising while simultaneously computing and storing the Key and Value representations for the Uncached Historical Actions. After denoising, the Target Actions generated in the `AR-k` are applied to the environment and serve as the Uncached Historical Actions in the `AR-k+1` step, which are then used to generate future Target Actions. (b) In current autoregressive inference step, the Key and Value representations for the Cached Historical Actions can be directly used by the Attention Computation. But for the Uncached Historical Actions and Denoising Targets, we need utilize a `QKV_MLP` layer to extract their Query, Key and Value representations. During Attention Computation, the Uncached Historical Actions are restricted to considering only actions within its own chunk (indicated by the purple line), while the Denoising Targets has access to the entire action sequence (indicated by the yellow line).

their Key and Value representations have been cached up to this step. Here, $l$ denotes the length of the Cached Historical Actions. During denoising, the Key and Value representations for the Cached Historical Actions $\tilde{\mathbf{A}}^k_{0:l-1}$ have been precomputed in prior autoregressive steps and can be directly used in attention computation. We denote them as $\mathbf{K}^{(0)}_{0:l-1}$ and $\mathbf{V}^{(0)}_{0:l-1}$, where $(\cdot)$ indicates the denoising timestep and $(0)$ means these Key and Value representations correspond to the final Target Actions generated in prior steps. For the Uncached Historical Actions $\tilde{\mathbf{A}}^k_{l:L-1}$ and Denoising Targets $\mathbf{N}^k$, we utilize a `QKV_MLP` layer to extract their Query, Key, and Value representations,

$$\{\mathbf{Q}^{(0)}_{l:L-1}, \mathbf{K}^{(0)}_{l:L-1}, \mathbf{V}^{(0)}_{l:L-1}\} = \texttt{QKV\_MLP}\ (\tilde{\mathbf{A}}^k_{l:L-1}) \tag{7}$$

$$\{\mathbf{Q}^{(t)}_{L:L+M-1}, \mathbf{K}^{(t)}_{L:L+M-1}, \mathbf{V}^{(t)}_{L:L+M-1}\} = \texttt{QKV\_MLP}\ (\mathbf{N}^k) \tag{8}$$

where $(t)$ indicates that those Query, Key and Value representations correspond to the Denoising Targets at timestep $t$. Then, $\mathbf{Q}^{(0)}_{l:L-1}, \mathbf{K}^{(0)}_{l:L-1}, \mathbf{V}^{(0)}_{l:L-1}$ are cached and will be shared across all denoising timesteps. Concatenating all the above representations forms the whole Query, Key, and Value representations (i.e. $\mathbf{Q}(k,t)$, $\mathbf{K}(k,t)$, and $\mathbf{V}(k,t)$) at denoising timestep $t$ in the `AR-k` step.

$$\mathbf{Q}(k,t) = \texttt{Concat}(\mathbf{Q}^{(0)}_{l:L-1},\ \mathbf{Q}^{(t)}_{L:L+M-1}) \tag{9}$$

$$\mathbf{K}(k,t) = \texttt{Concat}(\mathbf{K}^{(0)}_{0:l-1},\ \mathbf{K}^{(0)}_{l:L-1},\ \mathbf{K}^{(t)}_{L:L+M-1}) \tag{10}$$

$$\mathbf{V}(k,t) = \texttt{Concat}(\mathbf{V}^{(0)}_{0:l-1},\ \mathbf{V}^{(0)}_{l:L-1},\ \mathbf{V}^{(t)}_{L:L+M-1}) \tag{11}$$

where `Concat` denotes the concatenation operation along temporal dimension. Subsequently, the causal temporal attention is computed as:

$$\mathbf{Q}(k,t) \times [\mathbf{K}(k,t) \times \mathbf{V}(k,t) + \tilde{\mathbf{M}}] \tag{12}$$

where $\tilde{\mathbf{M}}$ denotes the attention mask with dimensions $(L-l+M)\times(L+M)$, mirroring the training attention mask illustrated in Fig. 4(b). As shown in Fig. 3(b), this mask ensures that: 1) The attention

Table 1: Quantitative evaluation of our *CDP* and baseline methods on **simulation tasks**, highlighting the effectiveness of our approach.

| Method | Adroit | | DexArt | | MetaWorld | | | | RoboFactory | |
|---|---|---|---|---|---|---|---|---|---|---|
| | hammer | Pen | Laptop | Toilet | Bin Picking | Box Close | Disassemble | Reach | Lift Barrier | Place Food |
| CDP3 (Ours) | **100%** | **68%** | **84%** | **68%** | **43%** | **54%** | **83%** | **39%** | **40%** | **19%** |
| DP3 [27] | **100%** | 49% | 81% | 65% | 38% | 42% | 69% | 24% | 35% | 13% |
| CDP (Ours) | **100%** | **37%** | **52%** | **38%** | **24%** | **35%** | **51%** | **38%** | 22% | **22%** |
| DP [7] | **100%** | 29% | 31% | 26% | 21% | 30% | 43% | 27% | **29%** | 17% |

weights for the Uncached Historical Actions are computed regardless of any other actions. 2) The prediction of Target Actions are based on all Historical Actions.

**Cyclic Temporal Embedding.** In traditional temporal embedding mechanisms, unique temporal embeddings are typically assigned to each action during the inference phase. However, this method is not effective due to the variable length of action sequences. To tackle this problem, we present the Cyclic Temporal Embedding mechanism. Instead of generating new embeddings, the Denoising Targets utilize temporal embeddings that are cyclically shifted from the beginning of the sequence. During the training process, a random cyclic offset is applied to the temporal embedding sequence of each sample. This randomization enables the model to generalize across various temporal offsets, thereby enhancing its robustness in the inference phase.

# 4 Simulation Experiments

## 4.1 Quantitative Results

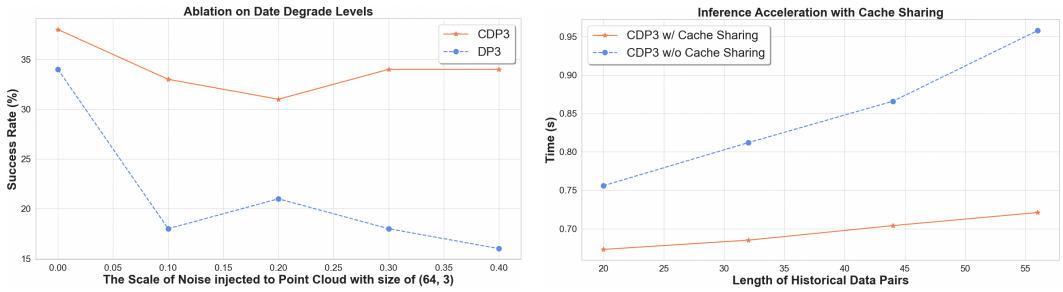
The quantitative results of all methods on the simulation benchmarks are detailed in Table 1, which illustrates that our *CDP* consistently outperforms the baseline methods across both 2D and 3D scenarios. Specifically, *CDP* achieves an improvement of about 5% to 20% points over the baseline methods across tasks of varying difficulty levels in different benchmarks. These findings highlight that, with an identical visual encoder, the causal generation paradigm employed by *CDP* significantly enhances the success rate of downstream tasks. This success can be attributed to two key aspects. First, the causal generation approach leverages historical actions by learning its temporal dynamic context to guide the generation of future actions. This temporal context provides valuable information that aids in making more informed and coherent decisions. Second, *CDP* effectively addresses the challenges caused by the degraded observations. In scenarios where the current observation alone cannot provide sufficient information for action generation, *CDP* utilizes the historical action sequence as a supplementary information. This dual reliance on both observations and historical actions ensures robustness in policy planning.

## 4.2 Ablation Study

In this section, we conduct experiments on the **Lift Barrier** task from RoboFactory [46] to rigorously evaluate both the effectiveness and efficiency of our proposed *CDP*.

**Robustness to Degraded Observation Data**. In this experiment, we introduce varying levels of noise to the low-resolution point clouds (size: $64 \times 3$), and assess the success rates of the compared methods when provided with these noisy inputs. Fig. 4a demonstrates that *CDP* exhibits markedly superior robustness to observation degradation. As the noise level increases, the success rate of DP3 declines precipitously, whereas our method maintains consistently high performance. This resilience is attributed to the model's ability to leverage historical action sequences, thereby compensating for the diminished spatial constraints inherent in degraded observations.

**Efficiency of Cache Sharing Mechanism**. We evaluate the efficiency of our Cache Sharing Mechanism on a single RTX 4090D by measuring per-AR-step inference latency while varying the length




(a) (b)

Figure 4: **Visualization of the Effectiveness and Efficiency of Our *CDP*.** (a) The performance of DP3 collapses as observation degrades, whereas our *CDP* remains nearly unaffected. (b) Our Cache Sharing Mechanism yields inference-time speed-ups that scale approximately linearly with the length of the Historical Actions.

Table 2: Quantitative results of our *CDP* and the baseline method on **real-world** tasks. Succ. is the abbreviation for Success Rate.

| Method | Collecting Objects | | | Stacking Cubes | | | Push T |
|---|---|---|---|---|---|---|---|
| | Grasping Succ. | Placing Succ. | Succ. | Grasping Succ. | Placing Succ. | Succ. | Succ. |
| CDP (Ours) | **18 / 20** | **13 / 18** | **13 / 20** | **16 / 20** | **10 / 16** | **10 / 20** | **17 / 20** |
| DP | **18 / 20** | 11 / 18 | 11 / 20 | 13 / 20 | 7 / 13 | 7 / 20 | 14 / 20 |

of the Historical Data Pairs (i.e. Historical Actions $\tilde{\mathbf{A}}$ and Observations $\mathbf{O}$). As shown in Fig. 4b, enabling Cache Sharing Mechanism consistently reduces latency, and this acceleration becomes more pronounced as the Historical Data Pairs lengthens.

## 5 Real-World Experiments

We list the quantitative results of our real-world experiments in Table 2. For the tasks of Collecting Objects and Stacking Cubes, we report not only the overall success rate but also the individual success rates for grasping and placing. The placing success rate is measured conditional on successful grasping. As shown in the table, our model outperforms the baseline method in terms of grasping, placing and overall success rate, highlighting the effectiveness of our approach. Figure 5 illustrates representative trajectories of real-world tasks generated by *CDP*. Consistent with simulation results, real-world experiments confirm that *CDP* attains high success rates across all tasks, even when trained with a modest number of 50 demonstrations.

## 6 Conclusion

Our proposed *CDP* effectively counteracts the impact of observation-quality degradation—arising from sensor noise, occlusions, and hardware limitations—on reliable robotic manipulation. Built upon a causal-transformer diffusion framework, the method captures critical temporal dependencies, thereby compensating for the spatial constraints diminished by degraded observations and preserving task performance. Extensive evaluations demonstrate that our *CDP* consistently outperforms existing approaches, attesting to its robustness and effectiveness. To facilitate real-time inference, we further introduce a Cache Sharing Mechanism that reuses pre-computed key–value attention tensors, yielding substantial reductions in the computational burden of autoregressive action generation without compromising accuracy.

**Limitation.** This study has not addressed tasks demanding extremely long horizons. The extension of our method to scenarios requiring sustained and coherent planning over extended temporal spans remains an avenue for future inquiry.

**Acknowledgments**

This work was partially supported by Shenzhen Science and Technology Program (JCYJ20220818103001002), the Guangdong Basic and Applied Basic Research Foundation (2022B1515020103, 2023B1515120087), the Guangdong Key Laboratory of Big Data Analysis and Processing, Sun Yat-sen University, China, and by the High-performance Computing Public Platform (Shenzhen Campus) of Sun Yat-sen University.

## References

[1] M. Shridhar, L. Manuelli, and D. Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023.

[2] C. Wang, L. Fan, J. Sun, R. Zhang, L. Fei-Fei, D. Xu, Y. Zhu, and A. Anandkumar. Mimicplay: Long-horizon imitation learning by watching human play. *arXiv preprint arXiv:2302.12422*, 2023.

[3] Y. Ze, G. Yan, Y.-H. Wu, A. Macaluso, Y. Ge, J. Ye, N. Hansen, L. E. Li, and X. Wang. Gnfactor: Multi-task real robot learning with generalizable neural feature fields. In *Conference on Robot Learning*, pages 284–301. PMLR, 2023.

[4] X. B. Peng, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, and S. Levine. Learning agile robotic locomotion skills by imitating animals. *arXiv preprint arXiv:2004.00784*, 2020.

[5] A. Agarwal, S. Uppal, K. Shaw, and D. Pathak. Dexterous functional grasping. *arXiv preprint arXiv:2312.02975*, 2023.

[6] S. Haldar, J. Pari, A. Rai, and L. Pinto. Teach a robot to fish: Versatile imitation from one minute of demonstrations. *arXiv preprint arXiv:2303.01497*, 2023.

[7] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

[8] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[9] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[10] M. Janner, Y. Du, J. B. Tenenbaum, and S. Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

[11] Y. Luo, C. Sun, J. B. Tenenbaum, and Y. Du. Potential based diffusion motion planning. *arXiv preprint arXiv:2407.06169*, 2024.

[12] J. Carvalho, A. T. Le, M. Baierl, D. Koert, and J. Peters. Motion planning diffusion: Learning and planning of robot motions with diffusion models. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1916–1923. IEEE, 2023.

[13] K. Saha, V. Mandadi, J. Reddy, A. Srikanth, A. Agarwal, B. Sen, A. Singh, and M. Krishna. Edmp: Ensemble-of-costs-guided diffusion for motion planning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10351–10358. IEEE, 2024.

[14] E. Zhou, Y. Qin, Z. Yin, Y. Huang, R. Zhang, L. Sheng, Y. Qiao, and J. Shao. Minedreamer: Learning to follow instructions via chain-of-imagination for simulated-world control. *arXiv preprint arXiv:2403.12037*, 2024.

[15] Y. Qin, A. Sun, Y. Hong, B. Wang, and R. Zhang. Navigatediff: Visual predictors are zero-shot navigation assistants. *arXiv preprint arXiv:2502.13894*, 2025.

[16] S. Huang, Z. Wang, P. Li, B. Jia, T. Liu, Y. Zhu, W. Liang, and S.-C. Zhu. Diffusion-based generation, optimization, and planning in 3d scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16750–16761, 2023.

[17] T. Pearce, T. Rashid, A. Kanervisto, D. Bignell, M. Sun, R. Georgescu, S. V. Macua, S. Z. Tan, I. Momennejad, K. Hofmann, et al. Imitating human behaviour with diffusion models. *arXiv preprint arXiv:2301.10677*, 2023.

[18] H. Ha, P. Florence, and S. Song. Scaling up and distilling down: Language-guided robot skill acquisition. In *Conference on Robot Learning*, pages 3766–3777. PMLR, 2023.

[19] Z. Xian, N. Gkanatsios, T. Gervet, T.-W. Ke, and K. Fragkiadaki. Chaineddiffuser: Unifying trajectory diffusion and keypose prediction for robotic manipulation. In *7th Annual Conference on Robot Learning*, 2023.

[20] X. Li, V. Belagali, J. Shang, and M. S. Ryoo. Crossway diffusion: Improving diffusion-based visuomotor policy via self-supervised learning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 16841–16849. IEEE, 2024.

[21] L. Wang, J. Zhao, Y. Du, E. H. Adelson, and R. Tedrake. Poco: Policy composition from and for heterogeneous robot learning. *arXiv preprint arXiv:2402.02511*, 2024.

[22] K. Chen, E. Lim, K. Lin, Y. Chen, and H. Soh. Don't start from scratch: Behavioral refinement via interpolant-based policy diffusion. *arXiv preprint arXiv:2402.16075*, 2024.

[23] K. Sridhar, S. Dutta, D. Jayaraman, J. Weimer, and I. Lee. Memory-consistent neural networks for imitation learning. *arXiv preprint arXiv:2310.06171*, 2023.

[24] T. Z. Zhao, J. Tompson, D. Driess, P. Florence, K. Ghasemipour, C. Finn, and A. Wahid. Aloha unleashed: A simple recipe for robot dexterity. *arXiv preprint arXiv:2410.13126*, 2024.

[25] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. *arXiv preprint arXiv:2402.10329*, 2024.

[26] L. Kang, X. Song, H. Zhou, Y. Qin, J. Yang, X. Liu, P. Torr, L. Bai, and Z. Yin. Viki-r: Coordinating embodied multi-agent cooperation via reinforcement learning. *arXiv preprint arXiv:2506.09049*, 2025.

[27] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

[28] L. Fu, H. Huang, G. Datta, L. Y. Chen, W. C.-H. Panitch, F. Liu, H. Li, and K. Goldberg. In-context imitation learning via next-token prediction. *arXiv preprint arXiv:2408.15980*, 2024.

[29] X. Zhang, Y. Liu, H. Chang, L. Schramm, and A. Boularias. Autoregressive action sequence learning for robotic manipulation. *IEEE Robotics and Automation Letters*, 2025.

[30] Z. Gong, P. Ding, S. Lyu, S. Huang, M. Sun, W. Zhao, Z. Fan, and D. Wang. Carp: Visuomotor policy learning via coarse-to-fine autoregressive prediction. *arXiv preprint arXiv:2412.06782*, 2024.

[31] J. Chen, J. Yu, C. Ge, L. Yao, E. Xie, Y. Wu, Z. Wang, J. Kwok, P. Luo, H. Lu, et al. Pixart-$\alpha$: Fast training of diffusion transformer for photorealistic text-to-image synthesis. *arXiv preprint arXiv:2310.00426*, 2023.

[32] W. Peebles and S. Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

[33] Y. Qin, Z. Shi, J. Yu, X. Wang, E. Zhou, L. Li, Z. Yin, X. Liu, L. Sheng, J. Shao, et al. Worldsimbench: Towards video generation models as world simulators. *arXiv preprint arXiv:2410.18072*, 2024.

[34] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[35] B. Chen, D. Martí Monsó, Y. Du, M. Simchowitz, R. Tedrake, and V. Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2024.

[36] T. Yin, Q. Zhang, R. Zhang, W. T. Freeman, F. Durand, E. Shechtman, and X. Huang. From slow bidirectional to fast causal video generators. *arXiv preprint arXiv:2412.07772*, 2024.

[37] Y. Guo, C. Yang, A. Rao, Z. Liang, Y. Wang, Y. Qiao, M. Agrawala, D. Lin, and B. Dai. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. *arXiv preprint arXiv:2307.04725*, 2023.

[38] H. Lu, G. Yang, N. Fei, Y. Huo, Z. Lu, P. Luo, and M. Ding. Vdt: General-purpose video diffusion transformers via mask modeling. *arXiv preprint arXiv:2305.13311*, 2023.

[39] X. Ma, Y. Wang, G. Jia, X. Chen, Z. Liu, Y.-F. Li, C. Chen, and Y. Qiao. Latte: Latent diffusion transformer for video generation. *arXiv preprint arXiv:2401.03048*, 2024.

[40] W. Ren, H. Yang, G. Zhang, C. Wei, X. Du, W. Huang, and W. Chen. Consisti2v: Enhancing visual consistency for image-to-video generation. *arXiv preprint arXiv:2402.04324*, 2024.

[41] J. Yu, Y. Qin, H. Che, Q. Liu, X. Wang, P. Wan, D. Zhang, and X. Liu. Position: Interactive generative video as next-generation game engine. *arXiv preprint arXiv:2503.17359*, 2025.

[42] J. Yu, Y. Qin, H. Che, Q. Liu, X. Wang, P. Wan, D. Zhang, K. Gai, H. Chen, and X. Liu. A survey of interactive generative video. *arXiv preprint arXiv:2504.21853*, 2025.

[43] J. Yu, Y. Qin, X. Wang, P. Wan, D. Zhang, and X. Liu. Gamefactory: Creating new games with generative interactive videos. *arXiv preprint arXiv:2501.08325*, 2025.

[44] J. Yu, J. Bai, Y. Qin, Q. Liu, X. Wang, P. Wan, D. Zhang, and X. Liu. Context as memory: Scene-consistent interactive long video generation with memory retrieval. *arXiv preprint arXiv:2506.03141*, 2025.

[45] K. Gao, J. Shi, H. Zhang, C. Wang, J. Xiao, and L. Chen. Ca2-vdm: Efficient autoregressive video diffusion model with causal generation and cache sharing. *arXiv preprint arXiv:2411.16375*, 2024.

[46] Y. Qin, L. Kang, X. Song, Z. Yin, X. Liu, X. Liu, R. Zhang, and L. Bai. Robofactory: Exploring embodied agent collaboration with compositional constraints. *arXiv preprint arXiv:2503.16408*, 2025.

[47] 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. *arXiv preprint arXiv:1709.10087*, 2017.

[48] C. Bao, H. Xu, Y. Qin, and X. Wang. Dexart: Benchmarking generalizable dexterous manipulation with articulated objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21190–21200, 2023.

[49] 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 *Conference on robot learning*, pages 1094–1100. PMLR, 2020.

# A Simulation Experiments Details

## A.1 Experiment Setup

**Benchmarks.** 1) **Adroit** [47]: This benchmark employs a multi-fingered Shadow robot within the MuJoCo environment to perform highly dexterous manipulation tasks, including interactions with articulated objects and rigid bodies. It is designed to test advanced manipulation skills and coordination. 2) **Dexart** [48]: This dataset utilizes the Allegro robot within the SAPIEN environment to execute high-precision dexterous manipulation tasks, primarily focusing on articulated object manipulation. It emphasizes fine motor skills and adaptability in complex scenarios. 3) **Meta-World** [49]: This benchmark operates within the MuJoCo environment, using a robotic gripper to perform a diverse range of manipulation tasks involving both articulated and rigid objects. Tasks are categorized by difficulty levels: easy, medium, hard, and very hard. Its evaluation covers tasks from medium to very hard difficulty levels. 4) **RoboFactory** [46]: This is a benchmark for embodied multi-agent systems, focusing on collaborative tasks with compositional constraints to ensure safe and efficient interactions. The task challenges include designing effective coordination mechanisms to enable agents to work together seamlessly, and ensuring robustness against uncertainties and dynamic changes in the environment during long-term task execution.

**Baselines.** To systematically evaluate the performance of our proposed method *CDP*, we selected Diffusion Policy and 3D Diffusion Policy as the 2D and 3D baseline methods, respectively. Different from previous works, we employed an MLP as the visual encoder for both the 2D and 3D baseline methods in our experiments to ensure consistency. All methods were subjected to the same number of observation and inference steps, and were trained using an identical set of expert demonstrations as well as an equivalent number of training epochs.

**Evaluation Metric.** Following the established metrics [27], each experiment was conducted across three independent trials, utilizing seed values of 0, 1, and 2, in the Adroit, DexArt, and MetaWorld benchmarks. For each seed, the policy was evaluated over 20 episodes every 200 training epochs, and the mean of the top 5 success rates was computed. The final reported performance consists of the mean and standard deviation of these success rates across above three seeds. In the Robofactory benchmark, each experiment was executed using only one seed value (*i.e.* 0), and was evaluated over 100 episodes at the 300th training epoch. This comprehensive evaluation metrics ensures a fair comparison between our *CDP* and baseline methods, providing a clear assessment of the performance improvements achieved by our approach.

## A.2 Settings

In this part, we provide detailed descriptions of all settings in our simulation experiments, including those related to the training strategy, as well as the inputs (Observations $\mathbf{O}$ and Historical Actions $\tilde{\mathbf{A}}$) and outputs (Target Actions $\mathbf{A}$).

Table 3: **All Settings in Simulation Experiments.** $\mathcal{N}(0,1)$ denotes Gaussian noise characterized by a mean of 0 and a variance of 1. Following [7], the Target Actions $\mathbf{A}$ is composed of **Valid Parts** and **Redundant Parts**.

| Parameter | Value |
|---|---|
| Length of Historical Actions | 20 |
| Length of Target Actions (Valid Parts) | 8 |
| Length of Target Actions (Redundant Parts) | 4 |
| Noise in Perturbed Historical Actions | $\frac{1}{6} \times \mathcal{N}(0,1)$ |
| Size of Point Clouds | (512, 3) |
| Size of Images | (84, 84, 3) |
| Batch Size | 64 |
| Epoch | 3000 |

Specifically, we made the following changes in the simulation experiments conducted on the RoboFactory benchmark,

- In the 2D scenario, images were resized to (64, 64, 3) instead of (84, 84, 3). And in the 3D scenario, the size of point clouds was reduced to (128, 3), deviating from the standard configuration.
- Noise in Perturbed Historical Actions was increased from $\frac{1}{6} \times \mathcal{N}(0, 1)$ to $\frac{1}{2} \times \mathcal{N}(0, 1)$.
- The training epoch is set to 300 in RoboFactory benchmark.

### A.3 Ablation Study

In this section, we conduct experiments on the **Lift Barrier** task from RoboFactory [46] to systematically examine the impact of individual hyperparameters on the performance of our *CDP*.

**Noise Scale in Perturbed Historical Actions**. To justify our design choice on noise scale in Perturbed Historical Actions, we list the quantitative results og our *CDP* on different noise scales in Table 4. A small noise scale reduces robustness to accumulating prediction errors, whereas an excessive magnitude distorts the underlying structure of the Historical Actions and impedes convergence. Empirically, an intermediate value of 1/6 strikes an effective balance, yielding the highest success rate across all tasks.

Table 4: Quantitative results of our proposed method on different noise scales injected in Perturbed Historical Actions.

| Method | Noise Scale | | | | | |
|---|---|---|---|---|---|---|
| | 1/9 | 1/6 | 1/3 | 1 | 3 | 6 |
| CDP3 | 30% | **40%** | 36% | 25% | 25% | 23% |

**Chunk Size**. To justify our design choice on chunk size, we list the quantitative results of our *CDP* on different chunk sizes in Table 5. For simpler tasks, we recommend a Chunk Size of 8, and a larger size for more complex tasks.

Table 5: Quantitative results of our proposed method on different chunk sizes.

| Method | Chunk Size | | | |
|---|---|---|---|---|
| | 1 | 2 | 4 | 8 |
| CDP3 | 1% | 5% | 35% | **40%** |

**Length of the Historical Actions**. The first two columns of Table 6 presents the success rates of the model for different configurations of Historical Actions length. From this table, we find that a longer sequence of Historical Actions tends to be beneficial in enhancing model performance.

Table 6: Quantitative results of our proposed methods across different length of Historical Actions. The format "$*_1$ - $*_2$ - $*_3$" is used to denote the following: $*_1$ represents the length of Historical Actions. $*_2$ and $*_3$ represent the length of valid and redundant parts of Target Actions, respectively.

| Method | ? - 4 - 4 | | | ? - 4 - 8 | | | 8 - ? - 8 | | |
|---|---|---|---|---|---|---|---|---|---|
| | 4 | 8 | 16 | 4 | 8 | 16 | 2 | 4 | 8 |
| CDP3 | 0% | 0% | 36% | 0% | 6% | 37% | 0% | 6% | **33%** |
| CDP | 0% | 0% | **37%** | 0 | 0% | **39%** | 0% | 0% | **38%** |

**Length of the Denoising Targets**. The last column in Table 6 illustrates that longer Denoising Targets will provide higher gains for the model. For the whole table, we observe that the optimal length of Denoising Targets should be determined in conjunction with the length of Historical

Actions. Only well-matched combinations of these action lengths can effectively enhance model performance.

### A.4 Qualitative Results of Simulation Experiments

Fig. 5 visualize the results of our *CDP* across various tasks on the Adroit, Dexart, MetaWorld, and RoboFactory benchmarks, which demonstrate the effectiveness of our model in multiple simulation environments.

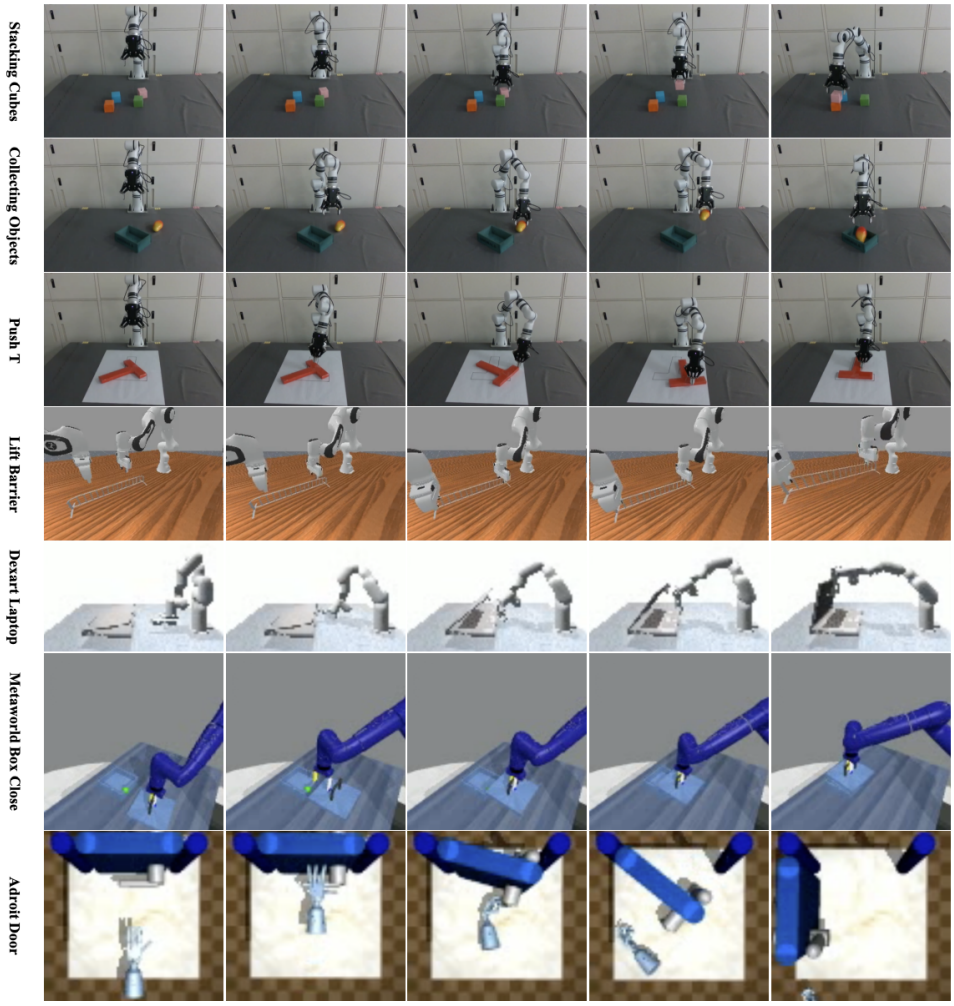



Figure 5: **Visualization of real-world and simulation experimental results,** including Stacking Cubes, Collecting Objects, Push T, Lift Barrier, Dexart Laptop, Metaworld Box Close and Adroit Door task from the top to the bottom.

## B Real-world Experiments Details

### B.1 Workspace

As illustrated in Fig. 6, our real-world experiments were conducted using a RealMan robotic arm fitted with a DH AG95 gripper. A stationary top-down Intel RealSense D435i RGB-D camera was employed to provide a global RGB image of the workspace. All hardware components were connected to a workstation equipped with an NVIDIA 3090 Ti GPU. This workspace enabled efficient data acquisition and evaluation of the robotic system's performance.

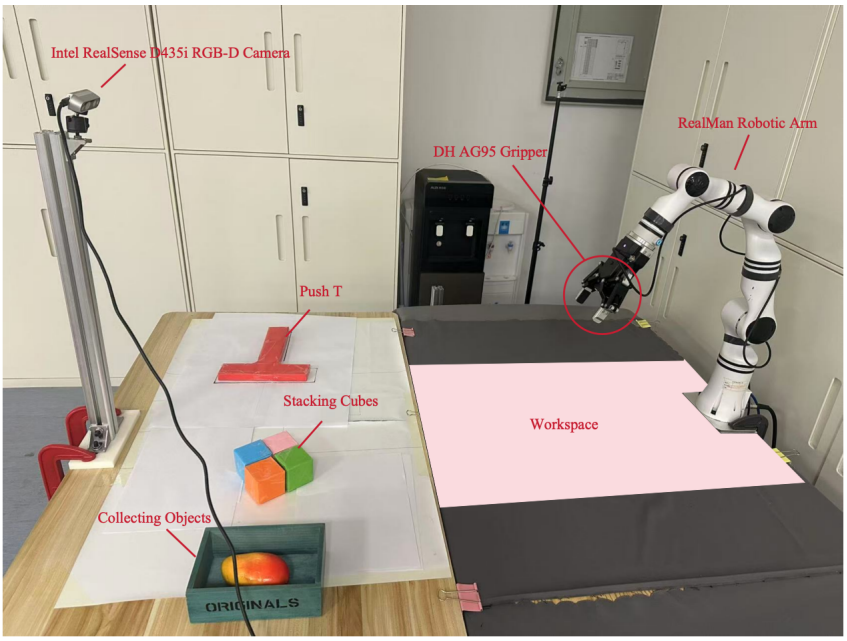



Figure 6: Real-world Experimental Workspace

### B.2 Experiment Setup

**Demonstrations.** The demonstrations utilized in our real-world experiments were meticulously collected via human teleoperation, facilitated by advanced vision-based retargeting techniques. For each task, we collected a total of 50 demonstrations, each carefully selected to encapsulate the fundamental skills and critical interactions necessary for successful task completion. This curation keeps the dataset compact while ensuring it faithfully represents the inherent complexities and challenges of each task.

**Settings.** All settings in our real-world experiments are list in Table 7. Similar with our simulation experiments, these settings involve the training strategy, as well as the inputs (Observations $\mathbf{O}$ and Historical Actions $\tilde{\mathbf{A}}$) and outputs (Target Actions $\mathbf{A}$).

Table 7: All Settings in Real-World Experiments.

| Parameter | Value |
|---|---|
| Length of Historical Actions | 8 |
| Length of Target Actions (Valid Parts) | 16 |
| Length of Target Actions (Redundant Parts) | 8 |
| Noise in Perturbed Historical Actions | $\frac{1}{2} \times \mathcal{N}(0, 1)$ |
| Batch Size | 64 |
| Size of Images | (60, 80, 3) |
| Epoch | 300 |

### B.3 Tasks

In real-world scenarios, we evaluate our proposed model and the baseline methods through three tasks: Collecting Objects, Stacking Cubes, and Push-T. The completion criteria for each task are described as follows:

- **Collecting Objects** is considered successful if the robot places the mango into the green box. This task not only assesses the ability to grasp the object but also the precision in placing it accurately into the designated green box.

- **Stacking Cubes** involves four colored cubes: blue, green, pink, and orange. This task is successful if the robot stacks the pink cube on top of the orange one while ignoring the blue and green cubes, which serve as distractors.
- **Push T**: The robot must rotate and translate a T-shaped block to position it within a specified range to achieve task success.