# H$^3$DP: Triply-Hierarchical Diffusion Policy for Visuomotor Learning

Yiyang Lu[1*],    Yufeng Tian[4*],    Zhecheng Yuan[1,2,3*],    Xianbang Wang[1],

Pu Hua[1,2,3],    Zhengrong Xue[1,2,3],    Huazhe Xu[1,2,3]

[1] Tsinghua University IIIS, [2] Shanghai Qi Zhi Institute, [3] Shanghai AI Lab, [4] Harbin Institute of Technology

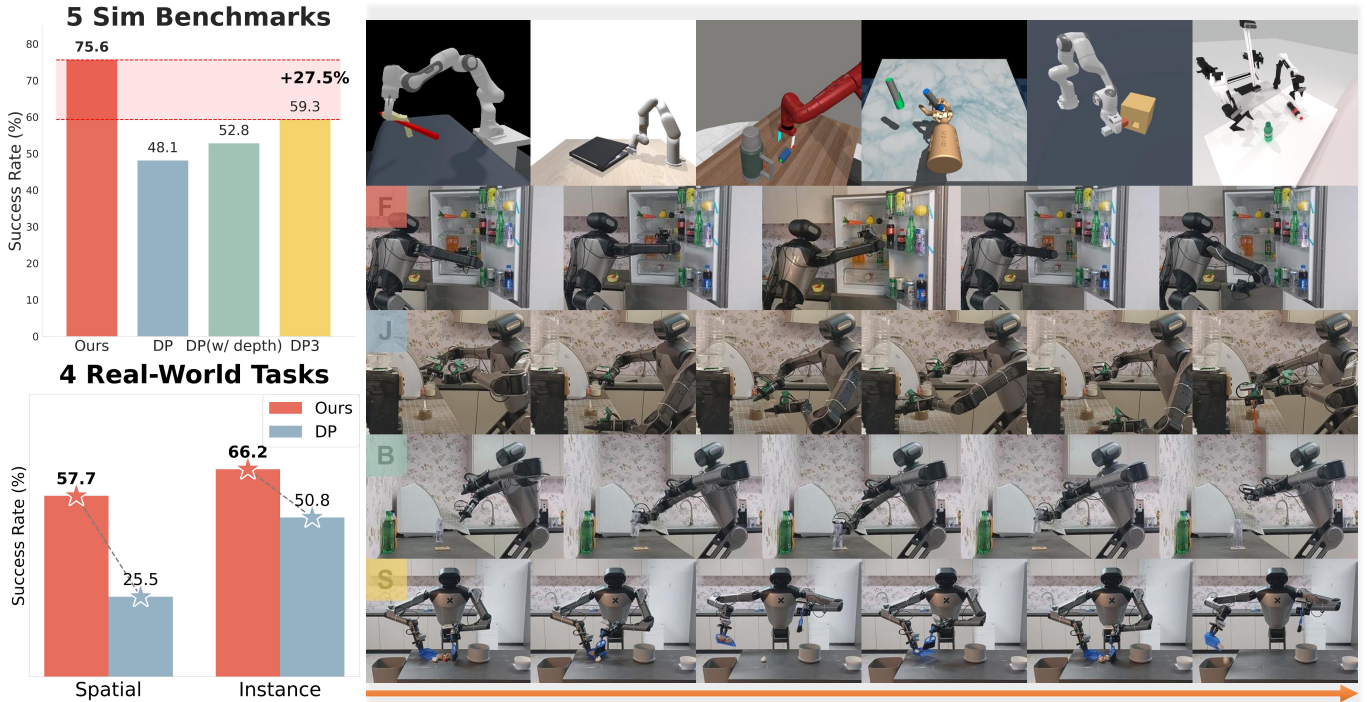luyy24@mails.tsinghua.edu.cn, huazhe_xu@mail.tsinghua.edu.cn

Fig. 1: **H$^3$DP** can not only achieve superior performance across 44 tasks on 5 simulation benchmarks, but also handle long-horizon challenging manipulation tasks in cluttered real-world scenarios.

*Abstract*—**Visuomotor policy learning has witnessed substantial progress in robotic manipulation, with recent approaches predominantly relying on generative models to model the action distribution. However, these methods often overlook the critical coupling between visual perception and action prediction. In this work, we introduce Triply-Hierarchical Diffusion Policy (H$^3$DP), a novel visuomotor learning framework that explicitly incorporates hierarchical structures to strengthen the integration between visual features and action generation. H$^3$DP contains 3 levels of hierarchy: (1) depth-aware input layering that organizes RGB-D observations based on depth information; (2) multi-scale visual representations that encode semantic features at varying levels of granularity; and (3) a hierarchically conditioned diffusion process that aligns the generation of coarse-to-fine actions with corresponding visual features. Extensive experiments demonstrate that H$^3$DP yields a $+27.5\%$ average relative improvement over baselines across 44 simulation tasks and achieves superior performance in 4 challenging bimanual real-world manipulation tasks. Project Page: https://lyy-iiis.github.io/h3dp/.**

## I. INTRODUCTION

Visuomotor policy learning has emerged as a prevailing paradigm in robotic manipulation [6, 65, 5, 62, 60]. Existing approaches have increasingly adopted powerful generative methods, such as diffusion and auto-regressive models, to model the action generation process [34, 57, 11, 45, 22]. However, these predominant methods have focused primarily on separately refining either the representation of perception or actions, often overlooking establishing a tight correspondence between perception and action. In contrast, human decision-making inherently involves hierarchical processing of information from perception to action [18, 3]. The visual cortex extracts features in a layered fashion and performs hierarchical inference based on visual motion perception, ultimately resulting in the generation of structured motor behaviors [23, 4]. Inspired by this, we argue that enabling learned visuomotor agents to emulate such hierarchical behavior patterns is also critical for

enhancing their decision-making capabilities.

Prior works have primarily focused on hierarchically modeling the action generation process alone [50, 13], without explicitly incorporating hierarchical structure throughout the whole visuomotor policy pipeline. In this paper, we present **H³DP**, a novel visuomotor policy learning framework grounded in three levels of hierarchy: input, representation, and action generation. This design reflects the hierarchical processing mechanisms that humans use the visual cortex to perceive environmental stimuli to guide motor behavior.

At the input level, to better leverage the depth information in modern robotic benchmarks and datasets [19, 27, 44, 12], H³DP moves beyond prior 2D approaches that primarily rely on RGB or simple RGB-D concatenation, which has shown limited effectiveness in prior work [62, 67]. We introduce **depth-aware layering** strategy that partitions the RGB-D input into distinct layers based on depth cues. This approach not only enables the policy to explicitly distinguish between foreground and background, but also suppresses distractors and occlusions [37, 1], thereby enhancing the understanding and reasoning of spatial structure in the cluttered visual scenarios.

For visual representation, to address the limitations of flattening image features into a single vector, which can discard some spatial structures and semantic information [15, 41, 25], H³DP employs **multi-scale visual representation**, where different scales capture features at varying granularity levels, ranging from global context to fine visual details.

In the action generation stage, H³DP incorporates a key inductive bias inherent to the diffusion process: the tendency to progressively reconstruct features from low-frequency to high-frequency components [40, 8, 55], by **hierarchical action generation**. Specifically, coarse visual features guide initial denoising steps to shape the global structure (low-frequency components) of action, while fine-grained visual features inform the later steps to refine precise details (high-frequency components). This establishes a tighter coupling between action generation and visual encoding, enabling the policy to generate actions that are semantically grounded in multi-scale perceptual features.

We validate H³DP through extensive experiments on **44** simulation tasks across **5** diverse benchmarks, where it surpasses state-of-the-art methods by a relative average margin of +**27.5%**. Furthermore, in real-world evaluations, we deploy bimanual robotic systems to tackle four challenging tasks situated in cluttered environments, involving high disturbances and long-horizon objectives. H³DP achieves a +**32.3%** performance improvement over Diffusion Policy in these real-world scenarios.

## II. RELATED WORK

**Visual imitation learning.** Numerous studies have proposed efficient policy learning algorithms from different aspects [6, 65, 58]. As a representative approach, to endow the learned policy multi-modality ability, Diffusion Policy [6] incorporates the diffusion process to better represent the action distribution. Based on DP, methods like DP3 [62, 61] and 3D-Actor [20],

designed for point cloud inputs, enhance the policy's scene understanding by refining the visual representation. Consistency Policy [34] and ManiCM [30] modify the inference process to achieve the inference acceleration. However, these approaches focus solely on enhancing either the action generation or the visual feature extraction, without explicitly modeling the relationship between them. To address this issue, we propose a hierarchical framework that couples multi-scale visual representations with the diffusion process, enabling a more structured integration between visual features and action generation.

**Leveraging hierarchical information for policy learning.** In the computer vision community, numerous studies have leveraged hierarchical information to address a variety of downstream tasks [53, 56, 24, 38, 29, 42]. For example, standard diffusion models [48, 17, 47, 49] and flow matching [26, 28, 10] adopt the U-Net framework [41, 66], which exploits multi-scale feature representations to retain rich contextual information throughout the denoising process. VAR [52] innovatively employs multi-scale visual representations with quantization to perform image generation in an auto-regressive manner. In robot learning, recent works [13, 33, 64] have also begun to adopt hierarchical paradigms for policy learning. Dense Policy [50] leverages a bidirectional extension strategy to enable hierarchical action prediction. ARP [64] predicts a sequence of actions at different levels of abstraction in a hierarchical way. CARP [13] draws inspiration from VAR by employing a multi-scale VQ-VAE [53, 38] to construct action sequences and subsequently generating residual actions autoregressively using a GPT-style architecture [35]. However, these algorithms model only the hierarchical structure of the action generation process, without explicitly addressing the crucial linkage between visual representation and action in visuomotor policy learning. In contrast, H³DP not only incorporates multi-scale visual representations but also leverages the inherent strengths of diffusion models to seamlessly integrate coarse-to-fine action generation into the diffusion process itself. Furthermore, by adopting a depth-aware layering strategy, H³DP maximizes the utilization of hierarchical feature information across the input, latent, and output stages, thereby enriching the policy learning pipeline in a structured and semantically aligned manner.

## III. METHOD

We employ three hierarchical structures to enhance the policy's understanding of visual input and predict more accurate action distributions. At the input level, the RGB-D image is discretized into multiple layers to improve the policy's ability to distinguish and interpret foreground-background variations. Upon this, we adopt a multi-scale visual representation, wherein coarse-grained features capture global contextual information, while fine-grained features encode detailed scene attributes. On the action side, correspondingly, the representations at different scales are utilized to generate actions in a coarse-to-fine manner, thus strengthening the correlation between action and visual representations. A detailed discussion of each part will be provided in the following sections.

## A. Depth-aware Layering

Effective robotic manipulation hinges on robust spatial understanding. While RGB data provides rich texture and color information, depth supplies the critical geometric context, including the relative spatial arrangement of objects and their distances. Combining these modalities offers a powerful foundation for scene comprehension. However, simply concatenating RGB images with depth maps does not lead to performance improvements [62, 67]. Hence, to fully exploit the geometric structure inherent in depth maps, we introduce a depth-aware layering mechanism inspired by Zhang et al. [63]. Pixels with depth $d$ are assigned to layer $m$ using linear-increasing discretization:

$$m = \left\lfloor -0.5 + 0.5\sqrt{1 + 4(N+1)(N+2)\frac{d - d_{\min}}{d_{\max} - d_{\min} + \epsilon}} \right\rfloor,$$
(1)

which promotes the robot to focus more on its workspace. By explicitly encoding objects distributed across different depth planes, this structured representation retains all visual detail while strategically utilizing depth to impose a meaningful foreground-background separation, thereby enabling the policy to selectively attend to different regions of the image. This design can effectively boost the agent's capacity for spatial perception and interaction planning. Furthermore, we also conduct comparisons against other discretization algorithms and perform additional experiments to substantiate the effectiveness of our proposed depth-aware layering method. The corresponding results are provided in Appendix E3 and Appendix E7.

## B. Multi-Scale Visual Representation

In visuomotor policy learning, visual representation plays a crucial role in embedding input images and mapping them to actions. An effective visual encoder should capture various granularity features of the visual scenarios and guide the policy to predict the action distribution. However, existing methods typically extract features at a single spatial scale or compress them into a fixed-resolution representation, limiting the expressiveness of learned features [15, 41, 25]. To address this problem, we hierarchically partition the feature map into multiple scales, enabling the capture of both coarse structural information and detailed local cues.

**Interpolation and Quantization.** After applying depth-aware layering to the input image $I$, each layer $I_m$ is independently encoded into multi-scale feature maps $\{f_{m,k}|f_{m,k} \in \mathbb{R}^{h_k \times w_k \times C}\}_{k=1}^K$, where $\{(h_k, w_k)\}_{k=1}^K$ denotes the spatial resolutions across scales. Adopting the quantization design in VQ-VAE [53, 38], these feature maps $\{f_{m,k}\}_{k=1}^K$ are quantized into discrete vectors drawn from a learnable codebook $\mathcal{Z}_m \in \mathbb{R}^{V \times C}$. Specifically, each feature vector $f_{m,k}^{(i,j)}$ is mapped to its nearest neighbor in Euclidean distance:

$$f_{m,k}^{(i,j)} \leftarrow \arg\min_{z \in \mathcal{Z}_m} \|z - f_{m,k}^{(i,j)}\|_2.$$
(2)

By applying differentiable interpolation and lightweight convolution to the quantized features $f_{m,k}$, we then obtain the

multi-scale visual representations $\{\hat{f}_{m,k}\}_{k=1}^K$ for each layer $I_m$. The pseudocode of full encoding procedure is detailed in Algorithm 1, Appendix B.

**Training.** To ensure consistent representations across scales, we aim to minimize the consistency loss between the original feature $f_m = \mathcal{E}_m(I_m)$ and the representation $\hat{f}_{m,k}$ at different scales:

$$\mathcal{L}_{\text{consistency}} = \sum_{m=0}^{N-1}\sum_{k=1}^{K}\left(\left\|\hat{f}_{m,k} - \text{sg}(f_m)\right\|_2^2 + \beta\left\|f_m - \text{sg}(\hat{f}_{m,k})\right\|_2^2\right),$$
(3)

where $\text{sg}(\cdot)$ is the stop gradient operator and $\beta$ balances the gradient flow between two terms. The visual encoder $\{\mathcal{E}_m\}_{m=0}^{N-1}$ and codebook $\{\mathcal{Z}_m\}_{m=0}^{N-1}$ are trained end-to-end, as described in detail in Appendix B.

## C. Hierarchical Action Generation

To match the inherent inductive biases of denoising process [40, 8, 55], we leverage multi-scale visual representations to model action generation in a coarse-to-fine manner. The early stage actions are derived from representations that capture global scene information, while fine-grained representations are responsible for generating detailed action components. This approach couples the visual representation and the action generation process via reinforcing their correspondence at the same hierarchical levels.

**Inference.** Our action generation module is a denoising diffusion model conditioned on multi-scale features $F = \{\hat{f}_k = \{\hat{f}_{m,k}\}_{m=0}^{N-1}\}_{k=1}^K$ and robot poses $q$. The denoising process unfolds over $T$ steps partitioned into $K$ stages $\cup_{k=1}^K(\tau_{k-1}, \tau_k]$. When $t \in (\tau_{k-1}, \tau_k]$, the denoising network $\epsilon_\theta^{(t)}$ conditioning on the corresponding feature map $\hat{f}_k$ and robot poses $q$, predicts the noise component

$$\epsilon^t = \epsilon_\theta^{(t)}(a^t|\hat{f}_k, q),$$
(4)

then generates $a^{t-1}$ from $a^t$ via:

$$a^{t-1} = \sqrt{\alpha_{t-1}}\left(\frac{a^t - \sqrt{1 - \alpha_t}\cdot\epsilon^t}{\sqrt{\alpha_t}}\right) + \sqrt{1 - \alpha_{t-1} - \sigma_t^2}\cdot\epsilon^t + \sigma_t\tilde{\epsilon}^t,$$
(5)

gradually transforming the Gaussian noise $a^T$ into the noise-free action $a^0$, where $\alpha_t, \sigma_t$ are fixed parameters depending on the noise scheduler, and $\tilde{\epsilon}^t \sim \mathcal{N}(0, \mathbf{I})$ is a Gaussian noise.

**Training.** To train the denoising network $\epsilon_\theta^{(t)}$, we randomly sample an observation-action pair $((I, q), a^0) \in \mathcal{D}$ and noise $\epsilon \sim \mathcal{N}(0, \mathbf{I})$. The network is optimized to predict $\epsilon$ given a noisy action conditioned on the final feature map $\hat{f}_K$ and robot pose $q$, via the objective:

$$\mathcal{L}_{\text{diffusion}} = \mathbb{E}_{a^0, \epsilon, t}\left[\gamma_t\|\epsilon_\theta^{(t)}(\sqrt{\alpha_t}a^0 + \sqrt{1 - \alpha_t}\epsilon|\hat{f}_K, q) - \epsilon\|^2\right],$$
(6)

where $\{\gamma_t\}$ are pre-defined coefficients. More implementation details can be found in Appendix A. By conditioning on the final feature $\hat{f}_K$ during training, gradients from the loss propagate through the entire hierarchical encoder, implicitly optimizing all $\{\hat{f}_k\}_{k=1}^K$. This design promotes consistency
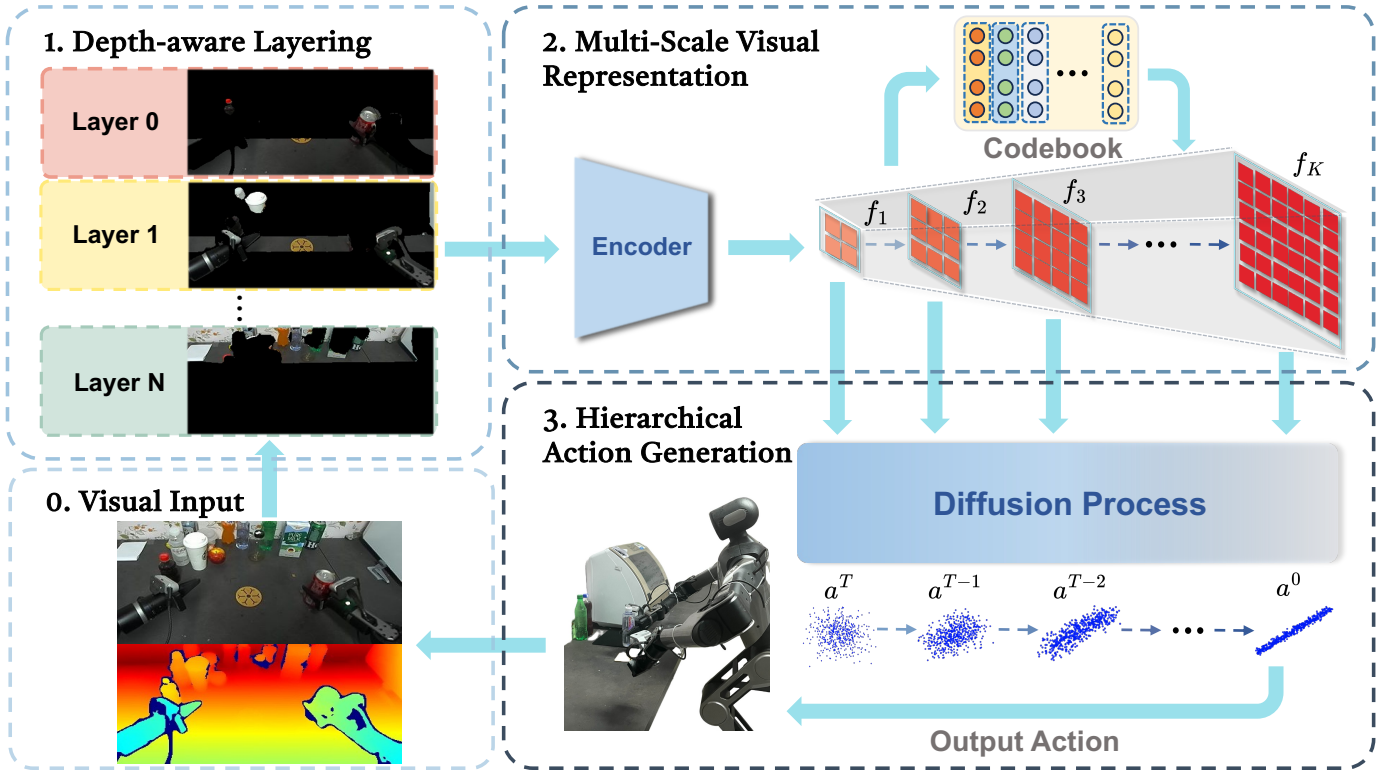
Fig. 2: **Overview of H³DP**. H³DP integrates three hierarchical design principles across the perception and action generation pipeline. At the input level, RGB-D images are decomposed into multiple layers based on their depth values. Then, we employ multi-scale visual representations to capture features at varying levels of granularity. During the action generation, denoising process is divided into several stages guided by multi-scale visual representations.

of representations at each scale for action generation while enhancing training efficiency.

**Discussions.** Diffusion models inherently aim to predict the posterior average of the target distribution conditioned on the provided features [7, 51], i.e., the optimal denoising network $\epsilon_{\theta*}^{(t)}$ follows $\epsilon_{\theta*}^{(t)}(a^t|f,q) = \mathbb{E}_{t,\epsilon,a^0,\sqrt{\alpha}_t a^0 + \sqrt{1-\alpha_t}\epsilon = a^t}[\epsilon|a^t, f, q]$. Features at varying resolutions retain information across distinct frequency domains. Consequently, they provide robust guidance for generating specific frequency components of the action during relevant stages of the denoising process. Related experiments are shown in Section IV-A3. By using lower-resolution features for earlier stages and gradually refining the predictions with higher-resolution features, the model benefits from both the stability of coarse representations and the precision of fine details.

## IV. EXPERIMENTS

In this section, we present extensive experiments across simulated and real-world settings to demonstrate the efficacy of H³DP. In addition, we perform thorough ablation analyses to evaluate the contribution of each hierarchical design.

### A. Simulation Experiments

*1) Experiment setup:* **Simulation benchmarks and baselines:** To sufficiently verify the effectiveness of H³DP, we evaluate H³DP on **5** simulation benchmarks, encompassing a total of **44** tasks. These tasks span a variety of manipulation

challenges, including articulated object manipulation [2, 36, 59], deformable object manipulation [14], bimanual manipulation [31], and dexterous manipulation [2, 36]. The details of the expert demonstrations can be found in Appendix C. To comprehensively assess the performance of H³DP, we compare it against three baselines: *Diffusion Policy* [6], one of the most widely used visuomotor policy learning algorithms; *Diffusion Policy (w/ depth)*, which extends Diffusion Policy to incorporate RGB-D input to bridge the information gap; and *DP3* [62], an enhanced version of Diffusion Policy that leverages an efficient encoder for point cloud input.

**Evaluation metric:** Each experiment is run with three different seeds to mitigate performance variance. For each seed, we evaluate 20 episodes every 200 training epoches. In simpler MetaWorld, Adroit and DexArt tasks, we compute the average of the highest five success rates as its success rate, while in other environments, only the hightest success rate is recorded.

*2) Simulation performance:* As shown in Table I, the simulation experiment results exhibit that H³DP outperforms or achieves comparable performance among the whole simulation benchmarks. Our method outperforms DP3 by a relative average margin of +**27.5**%. Notably, DP3 requires manual segmentation of the point cloud to remove background and task-irrelevant elements. This process introduces additional human effort and renders performance susceptible to segmentation quality. Relevant experimental results are provided in Appendix E6. In

TABLE I: **Simulation task results.** Across 5 simulation benchmarks with various difficult levels, H³DP obtains $+\mathbf{27.5\%}$ relative performance gains on average over 44 tasks.

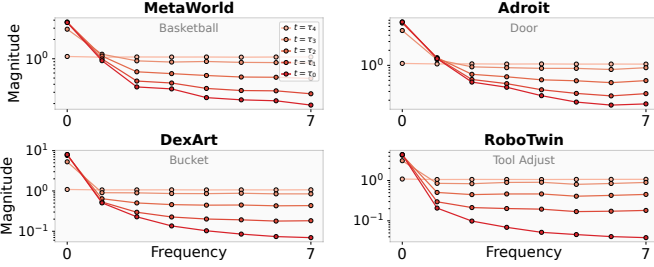| Method \ Tasks | MetaWorld (Medium 11) | MetaWorld (Hard 5) | MetaWorld (Hard++ 5) | ManiSkill (Deformable 4) | ManiSkill (Rigid 4) | Adroit (3) | DexArt (4) | RoboTwin (8) | **Average (44)** |
|---|---|---|---|---|---|---|---|---|---|
| **H³DP** | **98.3** | **87.8** | **95.8** | **59.3** | **65.3** | **87.3** | 53.3 | **57.4** | **75.6**±18.6 |
| DP | 78.2 | 52.6 | 58.0 | 22.3 | 27.5 | 79.0 | 44.3 | 22.8 | 48.1±23.1 |
| DP (w/ depth) | 77.7 | 57.2 | 71.2 | 44.5 | 40.8 | 76.0 | 42.0 | 12.6 | 52.8±22.2 |
| DP3 | 89.1 | 52.6 | 88.4 | 26.5 | 33.5 | 84.0 | 54.8 | 45.9 | 59.3±24.9 |



Fig. 3: **Action DFT results.** As the denoising process progresses, the Gaussian noise ($t = \tau_4$) is gradually transformed into the predicted action ($t = \tau_0$). Timesteps $\tau_i$ is arranged in descending order of noise levels. The results reveal a consistent frequency evolution pattern: low-frequency components predominantly emerge during the early stages of denoising, whereas high-frequency features are progressively introduced in the latter phases of the process.

contrast, benefiting from our design, H³DP obtains superior performance using only raw RGB-D input, without the need for segmentation and human effort. Furthermore, on the Adroit and DexArt benchmark, while DP3 leverages multi-view cameras to restore the complete point clouds, H³DP attains comparable performance using only one single-camera RGB-D image. The whole simulation results in each task can be found in Appendix E1.

*3) Spectral analysis of actions:* To gain a more comprehensive understanding of the action generation, we apply Discrete Fourier Transform (DFT) to examine how the frequency composition of actions evolves throughout the denoising process. Specifically, we conduct the analysis across 4 benchmarks and visualize the spectral characteristics of action chunks during generation. As shown in Figure 3, the results consistently indicate that the denoising process begins with the synthesis of low-frequency features, which are incrementally complemented by higher-frequency features in later stages. This observation not only shows that action, akin to image, exhibits an intrinsic inductive bias in the diffusion process, but also elucidates the action generation mechanism of H³DP, wherein actions are hierarchically composed to captured features across varying levels of granularity.

### B. Real-world Experiments

In terms of real-world experiments, we choose Galaxea R1 robot as our platform. We design four diverse challenging real-world tasks to evaluate the effectiveness of our method:

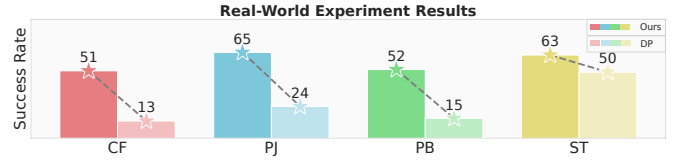**Clean Fridge (CF):** In a cluttered refrigerator environment,



Fig. 4: **Success rate in real-world.** Dark-colored bars correspond to H³DP, whereas the light-colored bars correspond to DP. H³DP outperforms DP in all 4 challenging real-world tasks.

the robot is required to relocate a transparent bottle from the upper compartment to the lower one. The bottle is randomized within a 30 cm × 5 cm region on both the upper and lower shelves of the refrigerator.

**Pour Juice (PJ):** This is a long-horizon task. The robot is required to place a cup in front of a water dispenser, scoop a spoonful of juice powder, then fill the cup with water, and finally put a straw in the cup. The cup is placed within a 7 cm × 7 cm area, and both the color of the juice powder and the position of the water dispenser are subject to variation across trials.

**Place Bottle (PB):** The robot must place a bottle, initially located at a random position, onto a designated coaster. The bottle is placed within a 15 cm × 15 cm region, while the coaster is positioned within an around 25 cm × 25 cm area.

**Sweep Trash (ST):** This long-horizon task entails picking up a broom, sweeping scattered debris on a table into a dustpan, and subsequently emptying the contents into a trash bin. The trash is randomly distributed across the entire table surface, approximately within a 40 cm × 40 cm area.

*1) Experiment Setup:* We use the ZED camera to acquire the depth image with 60Hz running frequency. The demonstrations are collected by Meta Quest3. Regarding



Fig. 5: **Experiment Setup.**

the two long-horizon tasks, both the baseline and our method incorporate the pre-trained ResNet18 [16] encoders for RGB modality to enhance the policy's perceptual capabilities in real-world environments. Each task is evaluated at 20 randomly sampled positions within the defined randomization range for each method. We record the success trials and calculate the corresponding success rate. In addition, during

policy deployment, we adopt an *asynchronous* design to obtain an approximately 15Hz inference speed. We also introduce *temporal ensembling* and *p-masking* to improve temporal consistency and alleviate overfitting to the proprioception state. More setup details can be found in Appendix D.

*2) Experiment Results:* **Spatial generalization:** As shown in Figure 4, H$^3$DP significantly outperforms the baseline across all four real-world tasks, achieving an average improvement of +**32.3**%. It should be noted that in **CF** and **PJ** tasks, the policy is required to not only identify target objects in cluttered visual environments but also perform long-horizon reasoning to accomplish the tasks. While DP struggles to complete either task, H$^3$DP achieves substantial improvements, outperforming DP by +**38**% and +**41**% respectively. Therefore, H$^3$DP demonstrates superior perceptual and decision-making capabilities compared to alternative algorithms. Meanwhile, it should be noted that in terms of the point cloud based method DP3, it requires precise segmentation and high-fidelity depth sensing, resulting in it being less effective in handling our four cluttered real-world scenes that we designed. Comparative experimental results for DP3 are presented in Appendix E8.

**Instance generalization:** Regarding instance generalization, we evaluate the model on two real-world tasks by varying the size and shape of bottles or trash items. As shown in Table II, after replacing the objects with variants of differing sizes and shapes, H$^3$DP maintains strong generalization capabilities attributable to its ability to hierarchically model features at multiple levels of granularity, and consistently outperforms baseline approaches across all settings.

TABLE II: **Instance generalization results.** H$^3$DP achieves +15.4% performance gain.

| Method \ Tasks | Place Bottle | | | Sweep Trash | | | Average |
| | coke bottle | sprite | can | 8 cm$^3$ | 64 cm$^3$ | 216 cm$^3$ | |
|---|---|---|---|---|---|---|---|
| **H$^3$DP** | **67** | **49** | **53** | **75** | **86** | **67** | **66.2** |
| Diffusion Policy | 45 | 36 | 40 | 52 | 72 | 60 | 50.8 |

*C. Ablation Study*

In this section, we ablate each key component of our framework and conduct experiments on three benchmarks to further exhibit the effectiveness of H$^3$DP. The entire results in each benchmark can be found in Appendix E2.

**Each hierarchical design.** We ablate the three hierarchical components introduced in our framework and compare them against the baseline Diffusion Policy with RGB-D input. As shown in Table III, each hierarchical component independently contributes to performance improvement, consistently outperforming the DP (w/ depth). Furthermore, Table III also demonstrates that the integration of all three hierarchical designs leads to a substantial enhancement in overall performance.

**The choice of $N$ in depth-aware layering.** For the depth-aware layering component, we investigate whether the policy's performance is sensitive to the choice of the number of layers $N$. As presented in Table IV, the trained policy achieves optimal and comparable performance when $N$ is set to 3 or 4, a trend consistently observed across all evaluated benchmarks. When $N$

TABLE III: **Ablation on hierarchical features.**

| Methods \ Benchmarks | MW | MS | RT | Average |
|---|---|---|---|---|
| **H$^3$DP** | **65.7** | **68.0** | **45.0** | **59.6** |
| w/o depth layering | 55.0 | 52.5 | 32.0 | 46.5 |
| w/o hierarchical action | 57.0 | 50.0 | 40.0 | 49.0 |
| w/o multi-scale representation | 53.7 | 52.5 | 40.0 | 48.7 |
| DP (w/ depth) | 46.7 | 47.5 | 32.0 | 42.1 |

becomes excessively large, the image is over-partitioned, thus reducing the representation capacity of the policy. Nevertheless, even in such cases, the performance remains slightly better than the non-layered baseline. These findings highlight the critical role of depth-aware layering in enhancing policy effectiveness.

TABLE IV: **Ablation on number of layers $N$.**

| Methods \ Benchmarks | MW | MS | RT | Average |
|---|---|---|---|---|
| H$^3$DP ($N = 1$) | 55.0 | 52.5 | 32.0 | 46.5 |
| H$^3$DP ($N = 2$) | 55.7 | 60.0 | 35.0 | 50.2 |
| H$^3$DP ($N = 3$) | 65.7 | **68.0** | 45.0 | **59.6** |
| H$^3$DP ($N = 4$) | **67.0** | 61.5 | **50.0** | **59.5** |
| H$^3$DP ($N = 5$) | 58.7 | 55.0 | **50.0** | 54.6 |
| H$^3$DP ($N = 6$) | 56.0 | 51.0 | 40.0 | 49.0 |

## V. Conclusion

In this paper, we introduce H$^3$DP, an efficient generalizable visuomotor policy learning framework that can obtain superior performance in a wide range of simulations and challenging real-world tasks. Extensive empirical evidence suggests that establishing a more cohesive integration between visual feature representations and the action generation process can enhance the generalization capacity and learning efficiency of our learned policies. The proposed three hierarchical designs not only facilitate the effective fusion of RGB and depth modalities, but also strengthen the correspondence between visual features and the generated actions at different granularity levels. In the future, we expect to extend the applicability of H$^3$DP to more intricate and fine-grained dexterous real-world tasks.

## VI. Limitations

Although H$^3$DP has demonstrated effectiveness in a variety of tasks, there exist several limitations. First, despite our use of asynchronous execution to improve inference speed in real-world settings, the overall inference time of diffusion-based models remains relatively slow. We could explore distilling the policy into a consistency model, to enhance real-time performance. Second, the limited depth quality of the ZED camera may hinder the policy's full potential in real-world deployment; employing higher-fidelity depth sensors could further boost the effectiveness of H$^3$DP in practical scenarios.

## References

[1] Stefan Ainetter, Christoph Böhm, Rohit Dhakate, Stephan Weiss, and Friedrich Fraundorfer. Depth-aware object segmentation and grasp detection for robotic picking tasks. *arXiv preprint arXiv:2111.11114*, 2021.

[2] Chen Bao, Helin Xu, Yuzhe Qin, and Xiaolong 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.

[3] Johannes Bill, Hrag Pailian, Samuel J Gershman, and Jan Drugowitsch. Hierarchical structure is employed by humans during visual motion perception. *Proceedings of the National Academy of Sciences*, 117(39):24581–24589, 2020.

[4] Johannes Bill, Samuel J Gershman, and Jan Drugowitsch. Visual motion perception as online hierarchical inference. *Nature communications*, 13(1):7403, 2022.

[5] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. $\pi_0$: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

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

[7] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[8] Sander Dieleman. Diffusion is spectral autoregression, 2024. URL https://sander.ai/2024/09/02/spectral-autoregression.html.

[9] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021.

[10] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

[11] Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. *arXiv preprint arXiv:2410.12557*, 2024.

[12] Haoran Geng, Feishi Wang, Songlin Wei, Yuyang Li, Bangjun Wang, Boshi An, Charlie Tianyue Cheng, Haozhe Lou, Peihao Li, Yen-Jen Wang, Yutong Liang, Dylan Goetting, Chaoyi Xu, Haozhe Chen, Yuxi Qian, Yiran Geng, Jiageng Mao, Weikang Wan, Mingtong Zhang, Jiangran Lyu, Siheng Zhao, Jiazhao Zhang, Jialiang Zhang, Chengyang Zhao, Haoran Lu, Yufei Ding, Ran Gong, Yuran Wang, Yuxuan Kuang, Ruihai Wu, Baoxiong Jia, Carlo Sferrazza, Hao Dong, Siyuan Huang, Koushil Sreenath, Yue Wang, Jitendra Malik, and Pieter Abbeel. Roboverse: Towards a unified platform, dataset and benchmark for scalable and generalizable robot learning, April 2025. URL https://github.com/RoboVerseOrg/RoboVerse.

[13] Zhefei Gong, Pengxiang Ding, Shangke Lyu, Siteng Huang, Mingyang Sun, Wei Zhao, Zhaoxin Fan, and Donglin Wang. Carp: Visuomotor policy learning via coarse-to-fine autoregressive prediction. *arXiv preprint arXiv:2412.06782*, 2024.

[14] Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, et al. Maniskill2: A unified benchmark for generalizable manipulation skills. *arXiv preprint arXiv:2302.04659*, 2023.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1904–1916, 2015.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[17] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[18] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106, 1962.

[19] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J. Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 2020.

[20] Tsung-Wei Ke, Nikolaos Gkanatsios, and Katerina Fragkiadaki. 3d diffuser actor: Policy diffusion with 3d scene representations. *arXiv preprint arXiv:2402.10885*, 2024.

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[22] Seungjae Lee, Yibin Wang, Haritheja Etukuru, H Jin Kim, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Behavior generation with latent actions. *arXiv preprint arXiv:2403.03181*, 2024.

[23] Tai Sing Lee and David Mumford. Hierarchical bayesian inference in the visual cortex. *Journal of the Optical Society of America A*, 20(7):1434–1448, 2003.

[24] Siyuan Li, Rui Wang, Minxue Tang, and Chongjie Zhang. Hierarchical reinforcement learning with advantage-based auxiliary rewards. *Advances in Neural Information Processing Systems*, 32, 2019.

[25] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

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

[27] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *arXiv preprint arXiv:2306.03310*, 2023.

[28] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

[29] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[30] Guanxing Lu, Zifeng Gao, Tianxing Chen, Wenxun Dai, Ziwei Wang, and Yansong Tang. Manicm: Real-time 3d diffusion policy via consistency model for robotic manipulation. *arXiv preprint arXiv:2406.01586*, 2024.

[31] Yao Mu, Tianxing Chen, Shijia Peng, Zanxin Chen, Zeyu Gao, Yude Zou, Lunkai Lin, Zhiqiang Xie, and Ping Luo. Robotwin: Dual-arm robot benchmark with generative digital twins (early version). *arXiv preprint arXiv:2409.02920*, 2024.

[32] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[33] Shubham Pateria, Budhitama Subagdja, Ah-hwee Tan, and Chai Quek. Hierarchical reinforcement learning: A comprehensive survey. *ACM Computing Surveys (CSUR)*, 54(5):1–35, 2021.

[34] Aaditya Prasad, Kevin Lin, Jimmy Wu, Linqi Zhou, and Jeannette Bohg. Consistency policy: Accelerated visuomotor policies via consistency distillation. *arXiv preprint arXiv:2405.07503*, 2024.

[35] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[36] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*, 2017.

[37] Deepak Rao, Quoc V Le, Thanathorn Phoka, Morgan Quigley, Attawith Sudsang, and Andrew Y Ng. Grasping novel objects with depth segmentation. In *2010 IEEE/RSJ international conference on intelligent robots and systems*, pages 2578–2585. IEEE, 2010.

[38] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

[39] Douglas A Reynolds et al. Gaussian mixture models. *Encyclopedia of biometrics*, 741(659-663):3, 2009.

[40] Severi Rissanen, Markus Heinonen, and Arno Solin. Generative modelling with inverse heat dissipation. *arXiv preprint arXiv:2206.13397*, 2022.

[41] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

[42] Chaitanya Ryali, Yuan-Ting Hu, Daniel Bolya, Chen Wei, Haoqi Fan, Po-Yao Huang, Vaibhav Aggarwal, Arkabandhu Chowdhury, Omid Poursaeed, Judy Hoffman, et al. Hiera: A hierarchical vision transformer without the bells-and-whistles. In *International conference on machine learning*, pages 29441–29454. PMLR, 2023.

[43] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[44] Carmelo Sferrazza, Dun-Ming Huang, Xingyu Lin, Youngwoon Lee, and Pieter Abbeel. Humanoidbench: Simulated humanoid benchmark for whole-body locomotion and manipulation. *arXiv Preprint arxiv:2403.10506*, 2024.

[45] Nur Muhammad Shafiullah, Zichen Cui, Ariuntuya Arty Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning $k$ modes with one stone. *Advances in neural information processing systems*, 35:22955–22968, 2022.

[46] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, pages 369–386. SPIE, 2019.

[47] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[48] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, pages 11895–11907, 2019.

[49] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[50] Yue Su, Xinyu Zhan, Hongjie Fang, Han Xue, Hao-Shu Fang, Yong-Lu Li, Cewu Lu, and Lixin Yang. Dense policy: Bidirectional autoregressive learning of actions. *arXiv preprint arXiv:2503.13217*, 2025.

[51] Qiao Sun, Zhicheng Jiang, Hanhong Zhao, and Kaiming He. Is noise conditioning necessary for denoising generative models? *arXiv preprint arXiv:2502.13129*, 2025.

[52] Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in neural information processing systems*, 37:84839–84865, 2024.

[53] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[54] Che Wang, Xufang Luo, Keith Ross, and Dongsheng Li. Vrl3: A data-driven framework for visual deep reinforcement learning. *Advances in Neural Information Processing Systems*, 35:32974–32988, 2022.

[55] Shuai Wang, Zhi Tian, Weilin Huang, and Limin Wang. Ddt: Decoupled diffusion transformer. *arXiv preprint arXiv:2504.05741*, 2025.

[56] Xin Wang, Geoffrey Oxholm, Da Zhang, and Yuan-Fang Wang. Multimodal transfer: A hierarchical deep convolutional neural network for fast artistic style transfer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5239–5247, 2017.

[57] Zhendong Wang, Zhaoshuo Li, Ajay Mandlekar, Zhenjia Xu, Jiaojiao Fan, Yashraj Narang, Linxi Fan, Yuke Zhu, Yogesh Balaji, Mingyuan Zhou, et al. One-step diffusion policy: Fast visuomotor policies via diffusion distillation. *arXiv preprint arXiv:2410.21257*, 2024.

[58] Zhengrong Xue, Shuying Deng, Zhenyang Chen, Yixuan Wang, Zhecheng Yuan, and Huazhe Xu. Demogen: Synthetic demonstration generation for data-efficient visuomotor policy learning. *arXiv preprint arXiv:2502.16932*, 2025.

[59] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR, 2020.

[60] Zhecheng Yuan, Tianming Wei, Shuiqi Cheng, Gu Zhang, Yuanpei Chen, and Huazhe Xu. Learning to manipulate anywhere: A visual generalizable framework for reinforcement learning. *arXiv preprint arXiv:2407.15815*, 2024.

[61] Yanjie Ze, Zixuan Chen, Wenhao Wang, Tianyi Chen, Xialin He, Ying Yuan, Xue Bin Peng, and Jiajun Wu. Generalizable humanoid manipulation with 3d diffusion policies. *arXiv preprint arXiv:2410.10803*, 2024.

[62] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

[63] Renrui Zhang, Han Qiu, Tai Wang, Ziyu Guo, Ziteng Cui, Yu Qiao, Hongsheng Li, and Peng Gao. Monodetr: Depth-guided transformer for monocular 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9155–9166, 2023.

[64] Xinyu Zhang, Yuhan Liu, Haonan Chang, Liam Schramm, and Abdeslam Boularias. Autoregressive action sequence learning for robotic manipulation. *IEEE Robotics and Automation Letters*, 2025.

[65] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[66] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: A nested u-net architecture for medical image segmentation. In *Deep learning in medical image analysis and multimodal learning for clinical decision support: 4th international workshop, DLMIA 2018, and 8th international workshop, ML-CDS 2018, held in conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, proceedings 4*, pages 3–11. Springer, 2018.

[67] Haoyi Zhu, Yating Wang, Di Huang, Weicai Ye, Wanli Ouyang, and Tong He. Point cloud matters: Rethinking the impact of different observation spaces on robot learning. *arXiv preprint arXiv:2402.02500*, 2024.

## A. Hyperparameters

To effectively address the varying levels of difficulty and distinct properties inherent to different benchmarks, we adapt our hyperparameter settings for each specific dataset. The chosen configurations, detailed in Table V, VI, VII, VIII, are selected based on previous works [6, 62, 67, 31].

TABLE V: **Hyperparameters used for MetaWorld, DexArt.**

| Hyperparameter | Value |
|---|---|
| Observation Horizon ($T_o$) | 2 |
| Action Horizon ($T_a$) | 2 |
| Prediction Action Horizon ($T_p$) | 4 |
| Optimizer | AdamW [21] |
| Betas ($\beta_1, \beta_2$) | [0.95, 0.999] |
| Learning Rate | 1.0e-4 |
| Weight Decay | 1.0e-6 |
| Learning Rate Scheduler | Cosine |
| Training Timesteps ($T$) | 50 |
| Inference Timesteps | 20 |
| Prediction Type | $\epsilon$-prediction |
| Image Resolution | $128 \times 128$ |
| Scale Number ($K$) | 4 |
| Multi-Scale Representation Resolutions ($\{(h_k, w_k)\}_{k=1}^{K}$) | $\{(1,1),(3,3),(5,5),(7,7)\}$ |
| Stage Boundaries ($\{\tau_k/T\}_{k=0}^{K}$) | $\{0,0.4,0.6,0.8,1.0\}$ |

TABLE VI: **Hyperparameters used for Adroit.**

| Hyperparameter | Value |
|---|---|
| Observation Horizon ($T_o$) | 2 |
| Action Horizon ($T_a$) | 2 |
| Prediction Action Horizon ($T_p$) | 4 |
| Optimizer | AdamW |
| Betas ($\beta_1, \beta_2$) | [0.95, 0.999] |
| Learning Rate | 1.0e-4 |
| Weight Decay | 1.0e-6 |
| Learning Rate Scheduler | Cosine |
| Training Timesteps ($T$) | 50 |
| Inference Timesteps | 20 |
| Prediction Type | $\epsilon$-prediction |
| Image Resolution | $84 \times 84$ |
| Scale Number ($K$) | 4 |
| Multi-Scale Representation Resolutions ($\{(h_k, w_k)\}_{k=1}^{K}$) | $\{(1,1),(3,3),(5,5),(6,6)\}$ |
| Stage Boundaries ($\{\tau_k/T\}_{k=0}^{K}$) | $\{0,0.4,0.6,0.8,1.0\}$ |

TABLE VII: **Hyperparameters used for ManiSkill.**

| Hyperparameter | Value |
|---|---|
| Observation Horizon ($T_o$) | 2 |
| Action Horizon ($T_a$) | 8 |
| Prediction Action Horizon ($T_p$) | 16 |
| Optimizer | AdamW |
| Betas ($\beta_1, \beta_2$) | [0.9, 0.95] |
| Learning Rate | 1.0e-4 |
| Weight Decay | 1.0e-4 |
| Learning Rate Scheduler | One Cycle LR [46] |
| Training Timesteps ($T$) | 100 |
| Inference Timesteps | 100 |
| Prediction Type | $\epsilon$-prediction |
| Image Resolution | $128 \times 128$ |
| Scale Number ($K$) | 4 |
| Multi-Scale Representation Resolutions ($\{(h_k, w_k)\}_{k=1}^{K}$) | $\{(1,1),(3,3),(5,5),(7,7)\}$ |
| Stage Boundaries ($\{\tau_k\}_{k=0}^{K}/T$) | $\{0,0.4,0.6,0.8,1.0\}$ |

TABLE VIII: **Hyperparameters used for RoboTwin.**

| Hyperparameter | Value |
|---|---|
| Observation Horizon ($T_o$) | 3 |
| Action Horizon ($T_a$) | 2 |
| Prediction Action Horizon ($T_p$) | 8 |
| Optimizer | AdamW |
| Betas ($\beta_1, \beta_2$) | [0.95, 0.999] |
| Learning Rate | 1.0e-4 |
| Weight Decay | 1.0e-6 |
| Learning Rate Scheduler | Cosine |
| Training Timesteps ($T$) | 100 |
| Inference Timesteps | 100 |
| Prediction Type | $\epsilon$-prediction |
| Image Resolution | $180 \times 320$ |
| Scale Number ($K$) | 4 |
| Multi-Scale Representation Resolutions ($\{(h_k, w_k)\}_{k=1}^{K}$) | $\{(1,3),(3,5),(5,7),(5,9)\}$ |
| Stage Boundaries ($\{\tau_k\}_{k=0}^{K}/T$) | $\{0,0.4,0.6,0.8,1.0\}$ |

In addition to the hyperparameters reported in the table, the choice of the number of layers $N$ demonstrates great importance, as shown in Table IV. Empirically, we choose $N = 4$ in Adroit, MetaWorld Hard and Hard++, and $N = 3$ in other benchmarks.

The noise scheduler for diffusion process is determined by $\alpha_t$, defined using function $f(t)$

$$\alpha_t = \frac{f(t)}{f(0)}, \quad \text{where} \quad f(t) = \cos^2\left(\frac{\pi}{2}\frac{t/T + s}{1 + s}\right). \tag{7}$$

Here, $T$ is the total number of diffusion timesteps and $s$ is an offset parameter.

For the reverse process, we employ different formulations depending on the environment. In MetaWorld, Adroit and DexArt, we follow the DDIM [47] approach, formulating the reverse process as an ODE, which corresponds to setting

$$\sigma_t = 0 \tag{8}$$

for all $t$. In ManiSkill and RoboTwin, we follow the design of DDPM [17] and formulate the reverse process as a Variance Preserving (VP) SDE [49]. In this case, for all $t$,

$$\sigma_t = \sqrt{\frac{1 - \alpha_{t-1}}{1 - \alpha_t}}\sqrt{1 - \frac{\alpha_t}{\alpha_{t-1}}}. \tag{9}$$

Furthermore, consider the weighting term $\gamma_t$ in Equation 6. Since the choice of $\gamma_t$ does not affect the optimal denoising network $\epsilon_{\theta^*}$, we set

$$\gamma_t = 1 \tag{10}$$

for all $t$.

### B. Method Details

This section outlines the implementation details of our multi-scale encoding. The encoder $\mathcal{E}_m$ for each depth layer $m$ adopts the architecture from VQGAN [9], ensuring strong representational capacity while preserving spatial information. We use interpolate to denote a differentiable resizing operation (e.g. bilinear or nearest-neighbor interpolation), which is crucial for enabling gradient flow during training. The function $\mathcal{Q}$ represents the quantization process detailed in Equation 2. Finally, after interpolating a feature map $f_{m,k}$ to the highest resolution, we apply a lightweight convolutional network $\phi_{m,k}$ designed to help restore fine details from the potentially lower-resolution source features.

The pseudocode for this process is outlined in Algorithm 1.

---

**Algorithm 1:** Multi-scale Encoding

---

1 **Inputs:** raw image $I$
2 **Hyperparameters:** depth layer number $N$, scale number $K$, resolutions $\{(h_k, w_k)\}_{k=1}^{K}$
3 Partition image $I$ into $N+1$ images $\{I_m\}_{m=0}^{N}$ according to Equation 1
4 **for** $m = 0, \ldots, N-1$ **do**
5     $f_m \leftarrow \mathcal{E}_m(I_m) \in \mathbb{R}^{h_K \times w_K \times C}$
6     **for** $k = 1, \ldots, K$ **do**
7         $f_{m,k} \leftarrow \text{interpolate}(f_m, h_k, w_k) \in \mathbb{R}^{h_k \times w_k \times C}$
8         $f_{m,k} \leftarrow \mathcal{Q}(f_{m,k})$
9         $f_{m,k} \leftarrow \phi_{m,k}(\text{interpolate}(f_{m,k}, h_K, w_K)) \in \mathbb{R}^{h_K \times w_K \times C}$
10         $\hat{f}_{m,k} \leftarrow \sum_{k' \leq k} f_{m,k'}$
11         $f_m \leftarrow f_m - \hat{f}_{m,k}$
12 **Return:** multi-scale features $F = \{\hat{f}_k = \{\hat{f}_{m,k}\}_{m=0}^{N-1}\}_{k=1}^{K}$

---

All trainable parameters, including the visual encoders $\{\mathcal{E}_m\}_{m=0}^{N-1}$, the codebooks $\{\mathcal{Z}_m\}_{m=0}^{N-1}$, the CNN parameters $\{\{\phi_{m,k}\}_{m=0}^{N-1}\}_{k=1}^{K}$, and the denoising network $\epsilon_\theta$, are trained jointly in an end-to-end manner. The optimization minimizes the combined objective function $\mathcal{L}$, defined as a weighted sum of consistency loss (Equation 3) and the diffusion loss (Equation 6):

$$\mathcal{L} = \mathcal{L}_{\text{diffusion}} + \alpha \mathcal{L}_{\text{consistency}}, \tag{11}$$

where $\alpha$ is a hyperparameter balancing the two loss terms.

### C. Expert Demonstrations

Regarding the MetaWorld [59] and the RoboTwin [31] benchmarks, we utilize scripted policies to generate expert demonstrations. In the case of ManiSkill [14] tasks, we employ the officially provided demonstrations. Trajectories for other simulation benchmarks are collected with agents trained by RL algorithms [62, 43, 54]. The expert policies are evaluated over 200 episodes, and their success rates are detailed in Table XX.

Given the varying difficulty levels across benchmarks, we provide a different number of demonstrations for each. Specifically, we provide 50 trajectories per task for MetaWorld, Adroit, and RoboTwin. For DexArt, we follow the setup in [62] and provide 100 trajectories per task. For ManiSkill, we use all official demonstrations: 1000 for rigid tasks and 200 for deformable tasks.

In real-world experiments, we collect demonstrations of varying quantity, depending on the complexity and horizon length of the tasks. For short-horizon tasks, the number of collected trajectories is relatively limited — 100 for Clean Fridge and 200 for Place Bottle. In contrast, long-horizon tasks demand more comprehensive data coverage. We collect more demonstrations: 270 for Pour Juice and 500 for Sweep Trash. These demonstrations play a crucial role in guiding the training process, especially in scenarios where exploration is challenging or unsafe.

### D. Real-world Training Details

As mentioned in [13], DP-based methods often suffer from low inference speed, which can cause the inference process to stall. Prior approaches, including DP3 [62], attempt to address this by increasing action horizon (e.g. $T_a = 4$ or $T_a = 8$) or reducing the number of model parameters (e.g. Simple DP3). However, these strategies often compromise manipulation accuracy and dexterity. A further complication is that increasing $T_a$ widens the temporal gap between consecutive inference steps, leading to greater discrepancies in observed information, and consequently, divergence in predicted actions. This often results in noticeable jitter and discontinuities in manipulation.

In general, DP-based methods are hindered by low inference speed, temporal inconsistency and overfitting to proprioceptive information. To address these challenges and improve real-world performance, we employ several empirical techniques.

*1) Higher Inference Speed:* To mitigate slow inference rooted in DP, we adopt an *asynchronous* design, achieving a final inference frequency of 10-15 Hz. Instead of waiting for the execution of all predicted actions before initiating the next inference cycle, our method performs inference concurrently with action execution. The predicted action

TABLE IX: **Comparison of real-world inference speeds for different methods.** The asynchronous version of our method demonstrates a significant speed-up by decoupling inference from action execution.

| Method | DP | DP3 | H$^3$DP | H$^3$DP (asynchronous) |
|---|---|---|---|---|
| Inference Speed (FPS) | 12.4 | 12.7 | 12.1 | **24.2** |

is stored in a queue to be executed at a fixed inference speed (10-15 Hz in practice, 12 Hz as average).

The inference speeds achieved in real-world scenarios are presented in Table IX. H$^3$DP (asynchronous) demonstrates a superior inference speed compared to standard DP [6] and DP3 [62], as well as our synchronous H$^3$DP implementation. In

addition to this speed advantage, H³DP features a shorter action sequence length ($T_a = 2$), which contributes to more dexterous manipulation capabilities.

*2) Temporal Consistency:* Having adopted the *asynchronous* design, we have obtained action sequences with overlapping time intervals. To ensure temporal smoothness and reduce discontinuities, we incorporate *temporal ensembling mechanism* from ACT [65]. As in ACT, H³DP performs a weighted average of actions with the same timestep across multiple overlapping sequences. This ensembling mitigates the gap between actions inferred from slightly different observations and effectively reduces jitter.

*3) Alleviate Overfitting:* Similar to other real-world robotic systems, H³DP is susceptible to overfitting on proprioceptive inputs, often neglecting the RGB-D information. This is evidenced by that the model generates similar actions regardless of variations in object positions. We hypothesize that this occurs because the simple, low-parameter MLP used to encode proprioception is easier to optimize than the more complex CNN used for RGB-D input, leading to reliance on the former.

To mitigate this, we introduce a $p$-masking strategy during training. This mechanism stochastically masks all proprioceptive inputs with probability $p$, which decays linearly over the training process. Specifically, for training timestep $t$ in a total horizon $T$, $p(t) = 1 - t/T$. This schedule encourages the model to rely more on RGB-D features early in training, helping it avoid early-stage overfitting and develop stronger visual grounding.

### E. Additional Experiment Results

*1) Simulation Results for Each Task:* We present the simulation results for each task in Table XIX, which serves as a supplement to Table I. For each experiment, we report the average success rate over three different random seeds. The final average result is obtained by averaging across all benchmarks.

We also provide the training progress of 4 algorithms on 12 various tasks across 3 different benchmarks in Figure 6. The selected tasks span a range of difficulties and are included without cherry picking to provide an unbiased view of each algorithm.

*2) The Whole Results of Ablation Study:* We present the entire results of our ablation study on each hierarchical design and number of layers $N$ in Table X and Table XI, as a supplement to Table III and Table IV. For each experiment, the success rate is reported by averaging over 3 different random seeds. The final average result is obtained by averaging across benchmarks.

TABLE X: **Whole results of ablation study on hierarchical features.**

| Method \ Tasks | MetaWorld | | | ManiSkill | | RoboTwin | Average |
| --- | --- | --- | --- | --- | --- | --- | --- |
| | Soccer | Stick Pull | Pick Out of Hole | Fill | Excavate | Tool Adjust | |
| **H³DP** | **85** | **75** | 37 | **98** | **38** | **45** | **59.6** |
| w/o depth layering | 59 | 72 | 34 | 78 | 27 | 32 | 46.5 |
| w/o hierarchical action | 64 | 67 | **40** | 82 | 18 | 40 | 49.0 |
| w/o multi-scale representation | 55 | 72 | 34 | 73 | 32 | 40 | 48.7 |
| DP (w/ depth) | 37 | 71 | 32 | 72 | 23 | 32 | 42.1 |

TABLE XI: **Whole results of ablation study on number of layers $N$.**

| Method \ Tasks | MetaWorld | | | ManiSkill | | RoboTwin | Average |
| --- | --- | --- | --- | --- | --- | --- | --- |
| | Soccer | Stick Pull | Pick Out of Hole | Fill | Excavate | Tool Adjust | |
| H³DP ($N = 1$) | 59 | 72 | 34 | 78 | 27 | 32 | 46.5 |
| H³DP ($N = 2$) | 64 | 70 | 33 | 85 | 35 | 35 | 50.2 |
| H³DP ($N = 3$) | **85** | 75 | 37 | **98** | **38** | 45 | **59.6** |
| H³DP ($N = 4$) | 78 | **83** | **40** | 90 | 33 | **50** | **59.5** |
| H³DP ($N = 5$) | 62 | 75 | 39 | 87 | 23 | **50** | 54.6 |
| H³DP ($N = 6$) | 61 | 73 | 34 | 77 | 25 | 40 | 49.0 |

*3) Comparison with a GMM-based Layering Variant:* To highlight the advantages of depth-aware layering, we conduct a comparison against a variant where this module is substituted with a classical foreground-background segmentation method, Gaussian Mixture Models (GMM) [39], named H³DP-GMM. As shown in Table XII, H³DP outperforms H³DP-GMM in all benchmarks. Notably, H³DP-GMM yields results comparable to a simple single-layer ($N = 1$) approach, further emphasizing the rationality and effectiveness of our proposed depth-aware layering strategy.

*4) Comparison with More Baselines:* Except diffusion-based algorithms, we also compare H³DP with the recent state-of-the-art method CARP [13], which uses multi-scale action VQ-VAE to build hierarchical action structures. Table XIII shows that H³DP outperforms CARP with an average improvement of 18.9%, indicating the importance of adopting hierarchical designs throughout visual features and action generation.

TABLE XII: **Comparison with GMM-based layering variant.** $H^3DP$ with depth-aware layering achieves superior performance compared to using GMM for layering.

| Method \ Tasks | MetaWorld | | | ManiSkill | | RoboTwin | Average |
|---|---|---|---|---|---|---|---|
| | Soccer | Stick Pull | Pick Out of Hole | Fill | Excavate | Tool Adjust | |
| **$H^3DP$** | **85** | **83** | **40** | **98** | **38** | **45** | **64.8** |
| $H^3DP$-GMM | 45 | 67 | 32 | 75 | 27 | 37 | 47.2 |
| $H^3DP$ ($N = 1$) | 59 | 72 | 34 | 78 | 27 | 32 | 50.3 |

TABLE XIII: **Comparison with CARP.** $H^3DP$ outperforms CARP with an average improvement of 18.9%.

| Method \ Tasks | MetaWorld | | | | | | | Average |
|---|---|---|---|---|---|---|---|---|
| | Box Close | Soccer | Stick Pull | Pick Out of Hole | Peg Insert Side | Hammer | Sweep | |
| $H^3DP$ | **98** | **85** | **83** | **40** | **98** | **100** | **100** | **86.3** |
| CARP | 82 | 53 | 71 | 15 | 69 | 82 | 100 | 67.4 |
| DP | 83 | 43 | 64 | 13 | 62 | 64 | 96 | 60.7 |

*5) Comparison with DP with Pre-trained Visual Encoder:* Prior work suggests that pre-trained visual representation may enhance spatial generalization of policy [58]. Hence, we investigate the impact of integrating a pre-trained visual encoder with the original DP. We specifically replace the standard ResNet encoder in DP with DINOv2 [32] model.

This variant, named DP-DINOv2, is evaluated on randomly selected tasks from the MetaWorld benchmark. The comparative results are presented in Table XIV. Although DP-DINOv2 shows a marginal improvement on some tasks compared to the original DP baseline, this comes with drawbacks, including a longer training time, inference latency and larger number of parameters($\sim$21M for DINOv2 with ViT-S) due to the DINOv2 architecture.

In contrast, $H^3DP$ utilizes an efficient visual encoder with less than 0.7M parameters, which achieves strong performance improvements over the original DP without incurring the aforementioned overheads.

TABLE XIV: **Comparison with DP with pre-trained visual encoder.** While DP-DINOv2 yields small improvement after paying additional cost, $H^3DP$ demonstrates superior performance.

| Method \ Tasks | MetaWorld | | | | | | Average |
|---|---|---|---|---|---|---|---|
| | Hand Insert | Pick Out of Hole | Disassemble | Stick Pull | Soccer | Sweep Into | |
| **$H^3DP$** | **100** | **40** | **96** | **83** | **85** | **100** | **84.0** |
| DP | 73 | 13 | 81 | 64 | 43 | 74 | 58.0 |
| DP-DINOv2 | 91 | 24 | 77 | 72 | 41 | 78 | 63.8 |

*6) Importance of Segmentation in DP3:* As highlighted in Section IV-A2, DP3 relies on manual segmentation of point cloud for optimal performance. To demonstrate this dependency, we evaluate DP3's performance under two distinct segmentation conditions using randomly selected tasks from the MetaWorld benchmark.

We compare the following two scenarios: *DP3 with ideal segmentation*, which utilizes clean segmented point clouds containing only the robot and task-relevant objects, as implemented in the original DP3 algorithm; *DP3 without ideal segmentation*, which utilizes point clouds that are intentionally processed to include desk surface upon which objects rest, while other background elements are still removed. This configuration simulates common real-world scenarios where simple or automated segmentation rules might fail to perfectly isolate the task-relevant objects.

As shown in Table XV, DP3's performance degrades substantially when operating on point clouds without ideal segmentation. This result confirms that DP3 is highly sensitive to the quality of the input point cloud segmentation.

TABLE XV: **Comparision of DP3 under different segmentation qualities.** We compare DP3 success rates on selected tasks when provided with different segmentation qualities, highlighting significant performance degradation.

| Method \ Tasks | MetaWorld | | | | | | Average |
|---|---|---|---|---|---|---|---|
| | Push | Shelf Place | Stick Pull | Soccer | Bin Picking | Pick Place Wall | |
| DP3 | **96** | **86** | **61** | **57** | **100** | **97** | **82.8** |
| DP3 (w/o ideal segmentation) | 89 | 26 | 48 | 29 | 50 | 84 | 54.3 |

In contrast, $H^3DP$ operates directly on raw image without requiring such pre-processing, thereby avoiding such failure mode and the associated need for careful, potentially manual, segmentation tuning, especially common in real-world scenarios.

*7) $H^3DP$ in Tasks with Significant Depth Variations:* As introduced in Section III-A, our depth-aware layering mechanism discretizes the depth map into distinct layers. This layering offers a crucial advantage in scenarios with significant depth

variations by providing a structured representation that preserves visual detail while emphasizing foreground-background separation. We will elaborate on this benefit and provide supporting comparative analysis here.

We conduct further experiments on tasks involving complex spatial arrangements, such as reaching for an object from closer to further or manipulating items in a cluttered scene, demanding a fine-grained understanding of relative object depth. Although raw RGB-D data contains this information implicitly, models may struggle to effectively utilize it, potentially treating the depth channel similarly to color channels or failing to prioritize significant depth discontinuities. Point cloud representations, inherently capturing 3D structures, often perform well in such scenarios as they directly encode geometric relationships.

Our depth-aware layering mechanism explicitly addresses this challenge for RGB-D inputs. By assigning pixels to discrete layers based on their depth values, we impose a structure that forces the model to differentiate between elements located at varying distances to camera. This discretization acts as an inductive bias, guiding the model to attend more strongly to the geometric layout and relative positioning of objects along the depth axis.

To empirically support our hypothesis, we conduct an ablation study focusing on tasks exhibiting significant depth variations. We compare the performance of three distinct approaches: DP3, DP (w/ depth) and H$^3$DP (only with depth-aware layering, i.e., without hierarchical action and multi-scale representation).

As seen in Table XVI, our observations reveal a consistent pattern: in tasks involving significant depth variations, point cloud–based policy initially demonstrated superior performance compared to standard RGB-D processing, represented by DP (w/ depth). However, upon integrating the depth-aware layering mechanism, H$^3$DP consistently outperforms the baseline on these tasks, which strongly supports our claim.

TABLE XVI: **Performance comparison demonstrating the effectiveness of depth-aware layering.** Tasks with significant depth variations show great improvement only with depth layering compared to DP (w/ depth), surpassing the point cloud baseline (DP3).

| Method \ Tasks | MetaWorld | | | | | | Average |
| | Push | Shelf Place | Disassemble | Soccer | Pick Place Wall | Peg Insert Side | |
|---|---|---|---|---|---|---|---|
| **H$^3$DP (only w/ depth layering)** | **100** | **95** | **98** | 55 | **100** | 86 | **89.0** |
| DP (w/ depth) | 79 | 29 | 76 | 37 | 80 | 53 | 59.0 |
| DP3 | 96 | 86 | **98** | **57** | 97 | **92** | 87.7 |

*8) Comparison with DP3 in Real-world Experiments:* DP3 [62] is a renowned baseline succeeding DP [6] in imitation learning and robotic manipulation, achieving state-of-the-art results in multiple simulation environments. However, DP3 has notable limitations. In particular, it relies heavily on high-quality point clouds, typically requiring precision sensors such as the RealSense L515 to function effectively.

In our setup, the head-mounted camera is a ZED, which produces relatively low-quality visual inputs. This hinders the direct application of DP3 in our experimental setting. To ensure a fair comparison, we evaluate both H$^3$DP and DP3 on two short-horizon real-world tasks both using the learning-from-scratch (LFS) encoder. The results are summarized in Table XVII.

It is evident that DP3 underperforms compared to H$^3$DP in both tasks, highlighting H$^3$DP's ability to robustly extract meaningful features from RGB-D inputs, even when the quality of visual input is suboptimal. Furthermore, we empirically find that employing spatially sparse convolution provides better performance than the DP3-style encoder, suggesting a promising direction for improving point cloud encoding in low-fidelity settings.

*9) Inference Speed:* As shown in Table XVIII, we evaluate the inference speed of different methods within simulated environments. The results indicate that the primary bottleneck of the inference speed of H$^3$DP lies in the diffusion process itself, whereas the additional operations introduced for processing visual inputs and managing multi-scale representations incur only minimal computational overhead. A corresponding analysis of inference speed in real-world scenarios is available in Appendix D1.

TABLE XVII: **Comparison of H$^3$DP and DP3 in real-world experiments.** We make comparison in 2 short-horizon real-world tasks and both use LFS encoders. H$^3$DP achieves +29.0% performance gain.

| Method \ Tasks | CF | PB | **Average** |
|---|---|---|---|
| **H$^3$DP** | **51** | **52** | **51.5** |
| DP3 | 12 | 33 | 22.5 |

TABLE XVIII: **Comparison of inference speeds for DP, DP3 and H$^3$DP in simulation tasks.** The result indicates that additional operations introduced in H$^3$DP are lightweight compared to the diffusion process.

| Method | DP | DP3 | H$^3$DP |
|---|---|---|---|
| Inference Speed (FPS) | 11.1 | 12.2 | 12.0 |

TABLE XIX: **Success rates on 44 simulation tasks**. Results of four different methods for each task are provided in this table. The summary across domains is shown in Table I.

| Method \ Tasks | MetaWorld [59] (Medium) | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Basketball | Bin Picking | Box Close | Coffee Pull | Coffee Push | Hammer | Soccer | Push Wall |
| **H³DP** | 100 | 100 | 98 | 100 | 100 | 100 | 85 | 100 |
| DP | 100 | 96 | 83 | 82 | 84 | 64 | 43 | 76 |
| DP (w/ depth) | 100 | 98 | 77 | 79 | 79 | 64 | 37 | 70 |
| DP3 | 100 | 100 | 78 | 100 | 100 | 97 | 57 | 95 |

| Method \ Tasks | MetaWorld (Medium) | | | MetaWorld (Hard) | | | | |
|---|---|---|---|---|---|---|---|---|
| | Peg Insert Side | Sweep | Sweep Into | Assembly | Hand Insert | Pick Out of Hole | Pick Place | Push |
| **H³DP** | 98 | 100 | 100 | 100 | 100 | 40 | 99 | 100 |
| DP | 62 | 96 | 74 | 100 | 73 | 13 | 0 | 77 |
| DP (w/ depth) | 53 | 98 | 100 | 100 | 75 | 32 | 0 | 79 |
| DP3 | 92 | 100 | 61 | 100 | 37 | 30 | 0 | 96 |

| Method \ Tasks | MetaWorld (Hard++) | | | | | DexArt [2] | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Shelf Place | Diassemble | Stick Pull | Stick Push | Pick Place Wall | Laptop | Faucet | Toilet | Bucket |
| **H³DP** | 100 | 96 | 83 | 100 | 100 | 81 | 34 | 70 | 28 |
| DP | 20 | 81 | 64 | 70 | 55 | 69 | 23 | 58 | 27 |
| DP (w/ depth) | 29 | 76 | 71 | 100 | 80 | 63 | 20 | 62 | 23 |
| DP3 | 86 | 98 | 61 | 100 | 97 | 80 | 33 | 79 | 27 |

| Method \ Tasks | Adroit [36] | | | ManiSkill [14] (Rigid) | | | |
|---|---|---|---|---|---|---|---|
| | Hammer | Door | Pen | Peg Insertion Side (Grasp) | Peg Insertion Side (Align) | Pick Cube | Turn Faucet |
| **H³DP** | 100 | 79 | 83 | 88 | 15 | 85 | 73 |
| DP | 95 | 69 | 73 | 78 | 7 | 17 | 8 |
| DP (w/ depth) | 100 | 66 | 62 | 93 | 12 | 33 | 23 |
| DP3 | 100 | 71 | 81 | 63 | 12 | 10 | 48 |

| Method \ Tasks | ManiSkill (Deformable) | | | | RoboTwin [31] | | |
|---|---|---|---|---|---|---|---|
| | Excavate | Hang | Pour | Fill | Apple Cabinet Storage | Dual Bottles Pick (Easy) | Dual Bottles Pick (Hard) |
| **H³DP** | 38 | 93 | 8 | 98 | 98 | 48 | 53 |
| DP | 2 | 52 | 0 | 36 | 73 | 53 | 28 |
| DP (w/ depth) | 23 | 78 | 7 | 72 | 2 | 33 | 25 |
| DP3 | 15 | 80 | 0 | 12 | 55 | 55 | 42 |

| Method \ Tasks | RoboTwin | | | | | **Average** |
|---|---|---|---|---|---|---|
| | Block Handover | Block Hammer Beat | Diverse Bottles Pick | Pick Apple Messy | Tool Adjust | |
| **H³DP** | 70 | 85 | 25 | 35 | 45 | **75.6±18.6** |
| DP | 28 | 0 | 0 | 0 | 0 | 48.1±23.1 |
| DP (w/ depth) | 0 | 0 | 2 | 7 | 32 | 52.8±22.2 |
| DP3 | 85 | 47 | 30 | 8 | 45 | 59.3±24.9 |

TABLE XX: **Success rates of experts on 44 simulation tasks.** We evaluate 200 episodes for each task. For ManiSkill tasks, the demonstrations are provided officially, and we record the success rates as 100%. The final average result is obtained by averaging across all benchmarks.

| Method \ Tasks | MetaWorld [59] (Medium) | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | Basketball | Bin Picking | Box Close | Coffee Pull | Coffee Push | Hammer | Soccer | Push Wall |
| Expert | 100.0 | 97.0 | 90.0 | 100.0 | 100.0 | 100.0 | 90.5 | 100.0 |

| Method \ Tasks | MetaWorld (Medium) | | | MetaWorld (Hard) | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | Peg Insert Side | Sweep | Sweep Into | Assembly | Hand Insert | Pick Out of Hole | Pick Place | Push |
| Expert | 92.0 | 100.0 | 90.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |

| Method \ Tasks | MetaWorld (Hard++) | | | | | DexArt [2] | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | Shelf Place | Diassemble | Stick Pull | Stick Push | Pick Place Wall | Laptop | Faucet | Toilet | Bucket |
| Expert | 99.5 | 92.5 | 95.0 | 100.0 | 99.5 | 86.5 | 58.0 | 66.5 | 80.0 |

| Method \ Tasks | Adroit [36] | | | ManiSkill [14] (Rigid) | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| | Hammer | Door | Pen | Peg Insertion Side (Grasp) | Peg Insertion Side (Align) | Pick Cube | Turn Faucet |
| Expert | 99.0 | 100.0 | 97.0 | 100.0 | 100.0 | 100.0 | 100.0 |

| Method \ Tasks | ManiSkill (Deformable) | | | | RoboTwin [31] | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| | Excavate | Hang | Pour | Fill | Apple Cabinet Storage | Dual Bottles Pick (Easy) | Dual Bottles Pick (Hard) |
| Expert | 100.0 | 100.0 | 100.0 | 100.0 | 96.0 | 97.0 | 55.5 |

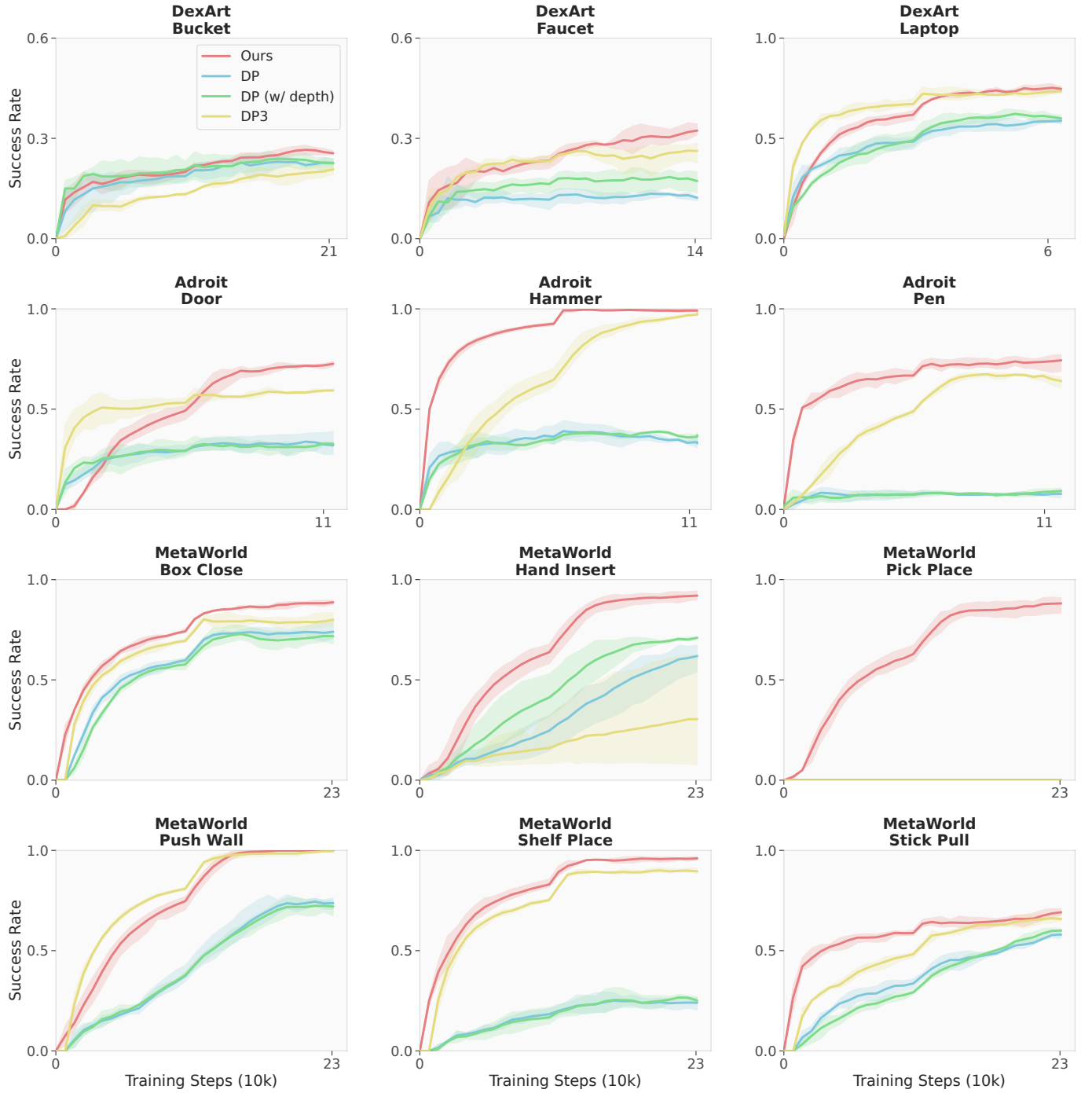| Method \ Tasks | RoboTwin | | | | | Average |
| --- | --- | --- | --- | --- | --- | --- |
| | Block Handover | Block Hammer Beat | Diverse Bottles Pick | Pick Apple Messy | Tool Adjust | |
| Expert | 98.0 | 97.0 | 72.0 | 88.5 | 86.5 | 93.9 |

Fig. 6: **Learning curves of the four methods on 12 randomly sampled diverse simulation tasks.** In most tasks, H³DP demonstrates faster convergence, higher final success rates, and lower variance compared to other three methods.