Prediction with Action:
Visual Policy Learning via Joint Denoising Process

Yanjiang Guo[12*], **Yucheng Hu**[13*], **Jianke Zhang**[1], **Yen-Jen Wang**[14], **Xiaoyu Chen**[12],
Chaochao Lu[3†], **Jianyu Chen**[12†]

[*]Equal Contribution [†]Corresponding Author
[1]IIIS, Tsinghua University [2]Shanghai Qizhi Institute
[3]Shanghai AI Lab [4]University of California, Berkeley
{guoyj22, huyc24}@mails.tsinghua.edu.cn

Abstract

Diffusion models have demonstrated remarkable capabilities in image generation tasks, including image editing and video creation, representing a good understanding of the physical world. On the other line, diffusion models have also shown promise in robotic control tasks by denoising actions, known as diffusion policy. Although the diffusion generative model and diffusion policy exhibit distinct capabilities—image prediction and robotic action, respectively—they technically follow a similar denoising process. In robotic tasks, the ability to predict future images and generate actions is highly correlated since they share the same underlying dynamics of the physical world. Building on this insight, we introduce **PAD**, a novel visual policy learning framework that unifies image **P**rediction and robot **A**ction within a joint **D**enoising process. Specifically, PAD utilizes Diffusion Transformers (DiT) to seamlessly integrate images and robot states, enabling the simultaneous prediction of future images and robot actions. Additionally, PAD supports co-training on both robotic demonstrations and large-scale video datasets and can be easily extended to other robotic modalities, such as depth images. PAD outperforms previous methods, achieving a significant 26.3% relative improvement on the full Metaworld benchmark, by utilizing a single text-conditioned visual policy within a data-efficient imitation learning setting. Furthermore, PAD demonstrates superior generalization to unseen tasks in real-world robot manipulation settings with 28.0% success rate increase compared to the strongest baseline. Project page at https://sites.google.com/view/pad-paper.
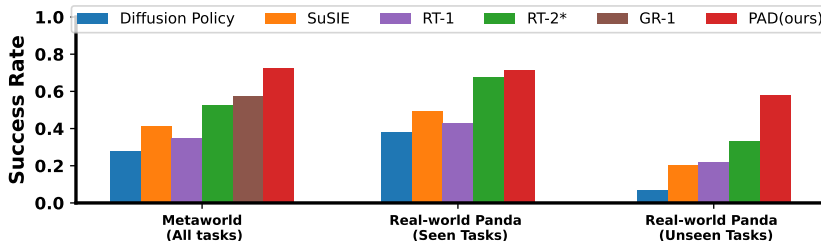
Figure 1: Multi-task performance comparisons in two domains.

1   Introduction

Making predictions and taking actions are critical human capabilities, allowing individuals to foresee the change of their surroundings and behave appropriately in response [1, 2]. Despite prediction and action seeming like two distinct abilities, they are highly coupled since they share the same
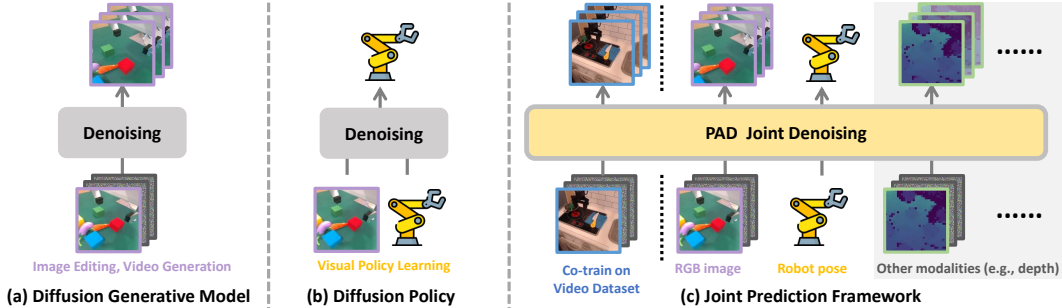
Figure 2: Diffusion models have achieved impressive success in visual generation tasks (a) and visual-motor control tasks (b). Image prediction and robot action are actually highly correlated since they share the same underlying physical dynamics. The PAD framework predicts the future and generates actions in a joint denoising process.

underlying physical laws of the world [3]. Understanding these laws enables humans to make better predictions and actions.

Recently, diffusion models [4, 5, 6] have achieved impressive success in visual generation tasks by training on extensive web-scale image and video datasets [7, 8, 9]. For example, image editing models can predict outcomes based on user instructions [10, 11, 12], while video generation models can generate sequences of future images [13, 14, 15], representing a good understanding of the physical world. On the other line, diffusion models have also shown efficacy in robotic control tasks by denoising actions conditioned on robot observations, known as diffusion policy [16]. Although the diffusion generative model and diffusion policy serve different functions across two domains, we believe that the capability for image prediction could significantly enhance robot policy learning, as they share the same fundamental physical laws. Previous works [17, 18, 19] have employed the image-editing model in an off-the-shelf manner by first synthesizing a goal image and subsequently learning a goal-conditioned policy. However, this two-stage approach separates the prediction and action learning process, neglecting deeper connections between prediction and action. In this way, actions do not leverage the pre-trained representations in the prediction models which encode rich knowledge of the physical world.

In this paper, we introduce the **P**rediction with **A**ction **D**iffuser (**PAD**), a unified policy learning framework that integrates prediction and action under the same diffusion transformer (DiT) architecture [20]. Specifically, we utilize the diffusion transformer model to seamlessly merge all modality inputs and simultaneously predict future images and actions via joint denoising, as illustrated in Figure 2(c). Additionally, the flexible DiT backbone also allows PAD to be co-trained on large-scale video data and extended to other robotic modalities, such as depth images. We have conducted extensive experiments on the MetaWorld Benchmark [21] as well as real-world robot arm manipulation tasks, demonstrating the efficacy of our approach, as shown in Figure 1. Our key contributions are:

- We propose a novel policy learning framework, Prediction with Action Diffuser (PAD), to predict futures and robot actions through a joint denoising process, benefiting policy learning for robotic tasks.

- The proposed PAD framework enables co-training of different datasets containing different modalities, allowing encoding rich physical knowledge from various data sources.

- We outperform previous methods with a clear margin in the Metaworld benchmark, surpassing baselines with a 26.3% relative improvement in success rate using a single visual-language conditioned policy. Furthermore, our method outperforms all baselines in the real-world robot manipulation experiments and can better generalize to unseen tasks.

2 Preliminaries

Problem Statement. We consider pixel-input language-conditioned robotic control under the imitation learning setting. We denote a robotic dataset $D_{robot} = \{\zeta_1, \zeta_2, ...\zeta_n\}$ comprising n demonstrations. The ith demonstration $\zeta_i = (I_i, l_i, \tau_i)$ contains a natural language instruction l_i, a

sequence of pixel inputs I_i, and a robot trajectory τ_i consisted of a sequence of robot poses $p_i^{1:T}$. However, since collecting robotic data is risky and costly, the scale of D_{robot} will be limited. We therefore also consider the RGB video dataset D_{video} which is easily accessible on the Internet. An instance in D_{video} can be represent as $\zeta_j = (I_j)$. Although D_{video} lacks robot action data, our proposed PAD framework enables co-training on both robotic dataset D_{robot} and video dataset D_{video}, leveraging the large-scale D_{video} data to enhance visual policy learning.

Latent Diffusion models. The core idea of diffusion models is to continuously add Gaussian noise to make a sample a Gaussian and leverage the denoising process for generating data [4]. Let $z_0 = \varepsilon(x_0)$ denote a latent sample encoded from real data. The noising process gradually adds normal Gaussian noise ($\mathcal{N}$) to z_0 over T steps, resulting in a set of noisy samples $Z = \{z_t | t \in [1, T]\}$, which is equivalent to sampling from the following distribution: $q(z_t | z_{t-1}) = \mathcal{N}(z_t; \sqrt{\alpha_t} z_{t-1}, (1 - \alpha_t)\mathbb{I})$, where $\{\alpha_t | t \in [1, T]\}$ are predefined hyper-parameters that control the amplitude of the noise. Let $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$, and according to DDPM [5], z_t can be directly obtained by adding a Gaussian noise ϵ_t to z_0: $z_t = \sqrt{\bar{\alpha}_t} z_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon_t$. Further, the denoising process starts with the most noisy latent sample z_T, and progressively reduces the noise to recover the real sample z_0 with condition c. It is based on a variational approximation of the probabilities $q(z_{t-1} | z_t, c)$ given by:

$$
\begin{aligned}
p(z_{t-1} | z_t, c) &= \mathcal{N}\left(z_{t-1}; \sqrt{\bar{\alpha}_{t-1}} \mu_\theta(z_t, t, c), (1 - \bar{\alpha}_{t-1})\mathbb{I}\right), \\
\mu_\theta(z_t, t, c) &= (z_t - \sqrt{1 - \bar{\alpha}_t} \epsilon_\theta(z_t, t, c)) / \sqrt{\bar{\alpha}_t}.
\end{aligned}
\tag{1}
$$

The noise estimator $\epsilon_\theta(z_t, t, c)$ is implemented as a neural network and is trained to approximate the gradient of the log-density of the distribution of noisy data [22]., that is:

$$
\epsilon_\theta(z_t, t, c) \approx -\sqrt{1 - \bar{\alpha}_t} \nabla_{z_t} \log p(z_t | c).
\tag{2}
$$

3 PAD: Prediction with Action via Joint Denoising Process

3.1 Overview of PAD

Multi-modalities Generation. In this section, we introduce our PAD framework, which concurrently predicts future frames and actions within a joint latent denoising process. We primarily focus on the RGB image modality M_I and the robot action modality M_A. Each robot action can be characterized by a robot pose that includes the position and rotation of the end-effector, as well as the gripper status. Notably, this framework can easily extend to extra modalities M_E. For instance, we additionally incorporate the depth image modality in the experiment part, which provides a more accurate measure of distances.

Conditional Generation. In the proposed PAD framework, predictions and actions are conditioned on multi-modality current observations, which include RGB images c_I, robot pose c_A, an additional depth map c_E (in Real-World tasks), and natural language instruction text l. The framework simultaneously outputs the corresponding future predictions x_I, x_E and robot action x_A. Rather than predicting a single future step, PAD can forecast k future steps $x_I^{1:k}, x_A^{1:k}, x_E^{1:k}$, which can be viewed as k step planning of the robot. Only the first predicted action x_A^1 is executed by the robot, which then triggers a new prediction cycle. This iterative prediction and execution process allows the robot to continuously plan and act in a closed-loop manner. The implementation details are discussed further in the subsequent section.

3.2 Model Architectures

Model Input Process. Given that the original data may come in various formats with high dimensions, we first map all modalities to a latent space and undertake a latent diffusion process. Following the process in [20], the RGB image x_I is initially processed through a pre-trained, frozen VAE[23] encoder ε_I to derive the latent representation $\varepsilon_I(x_I)$. This latent representation is then converted into a sequence of tokens t_I with embedding size h via tokenizer. Similarly, the robot pose x_A is encoded using a Multi-Layer Perceptron (MLP) [24] into $\varepsilon_A(x_A)$ and linearly transformed into tokens t_A with the same embedding size h. If available, the depth image is downsampled and tokenized into t_E. The natural language instruction is processed through a frozen CLIP encoder [25] to produce the text embedding c_l.
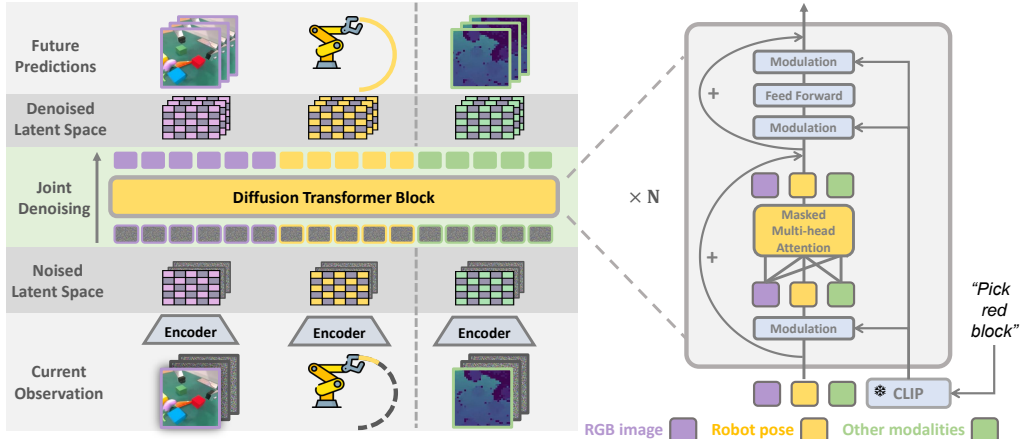
Figure 3: Visualization of the PAD framework. Current observations in different modalities are first encoded into latent and concatenated with white noise channel-wise. These noised latent are then tokenized into tokens and perform a joint denoising process to predict the images and robot actions simultaneously. PAD can flexibly accommodate extra or missing modal inputs through a masked-attention mechanism

Diffusion Transformer (DiT) Backbone. We have adopted the Diffusion Transformer (DiT) [20] as our model backbone, which offers several advantages over the U-net backbone commonly used in previous works [18, 17]. Notably, the DiT architecture efficiently integrates various modalities via the self-attention mechanism. Inputs such as RGB images, robot poses, and additional data are transformed into token sequences t_I, t_A, t_E with lengths T_I, T_A, T_E, respectively. These token sequences from different modalities are concatenated and undergo a joint latent denoising process

Furthermore, the DiT architecture is adaptable to missing modalities. For example, in the case of a video dataset that lacks robot actions, the input to DiT only comprises the image tokens t_I. We simply extend the token sequence to the combined length $T_I + T_A + T_E$ and introduce an attention mask in the self-attention block to exclude the padding tokens. Only effective predictions are retained in the output, discarding any padded parts. A brief illustration of the whole process is depicted on the right side of Figure 3. This design choice enables PAD to be concurrently trained on both RGB-only video datasets and robotic datasets.

Joint Conditional Generation. We initialize future observations as white noise and aim to reconstruct future observation frames and desired robot action, conditioning on current observations c_I, c_A, c_E. Following a similar strategy as in [26], we concatenate conditional latent and noise latent in the channel dimension. Specifically, after obtaining encoded latent $\varepsilon_I(c_I), \varepsilon_A(c_A), \varepsilon_E(c_E)$, we concatenate these latent with noise to obtain conditioned noised latent $L_I = [\varepsilon_I(c_I), z_t^I], L_A = [\varepsilon_A(c_A), z_t^A], L_E = [\varepsilon_E(c_E), z_t^E]$. For instance, if the encoded latent $\varepsilon_I(c_I)$ has a shape of $c \times d \times d$, then z_t^I would have a shape of $kc \times d \times d$ to represent k future frames, resulting in the final latent L_I having a shape of $(k+1)c \times d \times d$. The other modalities undergo a similar process.

After concatenating the latent, these conditioned noisy latent from different modalities are tokenized into sequences of tokens t_I, t_A, t_E with the same embedding size. The tokenization of image latent L_I follows a patchify process same to [20], while the tokenization of robot pose employs a simple linear projection. Finally, these tokens are fed into multiple layers of DiT to predict the latent representation of future frames. An illustration of the overall process can be found in Figure 3.

3.3 Training Process

Initialization. Following the initialization process in [15], we also initialize the PAD weights from the DiT model pre-trained on ImageNet for the image generation task conditioned on class [20]. However, we can not directly load the model since we have missing or incompatible model parameters. We discard the label embedding layers in DiT and zero-initialize new layers for text embedding, we replicate the weight of the image latent tokenizer for $k+1$ times to encode the stacked latent, and the encoder and decoder for robot state are also zero-initialized.

Training Objective. The diffusion process adds noise to the target encoded latent $\{\varepsilon_I(x_I), \varepsilon_A(x_A), \varepsilon_E(x_E)\}$ and results in noised latent $Z_{I,A,E} = \{z_t^I, z_t^A, z_t^E\}$. We train the PAD model to simultaneously predict the noise $\epsilon^I, \epsilon^A, \epsilon^E$ added to the sample data, conditioned on current observations $C_{I,A,E} = \{c_I, c_A, c_E\}$ and instructions l. This denoiser is trained with the DDPM [5] loss:

$$\mathcal{L}_{diff}^{\delta}(\theta) = \mathbb{E}_{\epsilon^\delta \sim \mathcal{N}(0,1), t, C, l} \left[\left|\left| \epsilon^\delta - \epsilon_\theta^\delta \left(z_t^\delta, t, C, l \right) \right|\right|_2^2 \right], \tag{3}$$

where $\delta \in \{I, A, E\}$ represents different types of input modalities. The denoising loss $\mathcal{L}_{diff}^{\delta}$ aims to maximize the evidence lower bound (ELBO) [5] while approximating the conditional distribution $p(\varepsilon_\delta(x_\delta)|C, l)$. We jointly minimize the following latent diffusion objectives and use hyperparameters $\lambda_I, \lambda_A, \lambda_E$ to balance the prediction loss between different modalities. Formally, the final training objective is given by:

$$\mathcal{L}(\theta) = \lambda_I \mathcal{L}_{diff}^I + \lambda_A \mathcal{L}_{diff}^A + \lambda_E \mathcal{L}_{diff}^E. \tag{4}$$

4 Experiments

In this section, we conduct a series of experiments on the simulated Metaworld Benchmark [21] and a real-world table manipulation suite, utilizing our joint prediction framework. We aim to answer the following questions:

- Can PAD enhance visual policy learning through joint prediction and action with limited robotic data?
- Can PAD benefit from co-training on large-scale internet video datasets and better generalize to unseen tasks?
- Can scaling up computational resources improve PAD's performance?
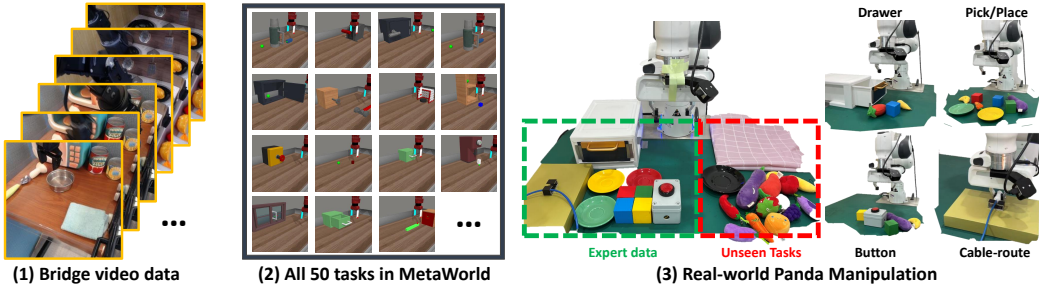

Figure 4: We learn a single vision-language conditioned policy to solve all tasks in each domain with limited demonstrations, co-training with the bridge video data. In simulated MetaWorld, we learn a policy to tackle all 50 tasks. In real-world panda manipulations, we split objects into seen objects and unseen new objects to test the generalization ability of our policy.

4.1 Environmental Setups and Baselines

Metaworld. Metaworld [21] serves as a widely used benchmark for robotic manipulation, accommodating both low-level feature and pixel input modalities. Previous studies that utilized pixel input generally developed separate task-specific policies for each of the 50 tasks. In contrast, our approach demonstrates a significant advancement by employing a single text-conditioned visual policy to address all 50 tasks, within a data-efficient imitation learning framework. We collected 50 trajectories per task, consistently using the "corner2" camera viewpoint and recording the robot's pose with 4-dimensional states that include end-effector position and gripper status. For a fair comparison, we do not utilize an additional depth input in Metaworld.

Real-World Panda Manipulation Tasks. Our real-world experiments involve a Panda arm performing diverse manipulation tasks such as pressing buttons, opening drawers, routing cables, and picking and placing with various objects, as shown in Figure 4. We follow the same hardware setup described in SERL [27] and utilize a wrist-mounted camera for pixel input [28]. The robot's poses are represented by 7-dimensional vectors, including 3 end-effector positions, 3 rotation angles, and 1

Easier Tasks	button-press	button topdown	drawer-open	door-open	faucet-close	plate-slide	reach-wall	window-open	window-close	door-lock
Diffusion Policy	0.92	0.16	0.36	0.32	0.76	0.60	0.72	0.60	0.36	0.12
SuSIE	0.96	0.32	0.60	0.68	0.56	0.68	0.92	0.68	0.96	0.32
RT-1	0.88	**1.00**	0.56	0.56	**1.00**	0.08	0.12	**1.00**	**1.00**	0.00
RT-2*	**1.00**	0.84	0.92	0.96	0.96	**0.88**	0.76	**1.00**	0.96	0.40
GR-1	**1.00**	0.84	**1.00**	**1.00**	0.96	**0.88**	**1.00**	**1.00**	**1.00**	0.60
PAD (ours)	**1.00**	0.92	**1.00**	**1.00**	0.92	0.72	**1.00**	0.92	**1.00**	**0.88**
PAD w/o img	**1.00**	0.92	**1.00**	0.88	0.92	0.16	0.92	**1.00**	**1.00**	0.12
PAD w/o co-train	**1.00**	0.92	**1.00**	0.92	0.92	0.48	0.92	0.96	0.96	0.72

Harder Tasks	assem-ble	basket-ball	coffee-pull	hammer	peg-insert	pick-place-wall	shelf-place	stick-push	stick-pull	Average (50tasks)
Diffusion Policy	0.20	0.08	0.00	0.08	0.16	0.36	0.00	0.00	0.00	0.279
SuSIE	0.40	0.24	0.32	0.04	0.24	0.24	0.08	0.16	0.16	0.410
RT-1	0.00	0.00	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.346
RT-2*	0.24	0.08	0.68	0.20	0.12	0.32	0.20	0.12	0.00	0.522
GR-1	0.64	0.08	0.52	0.48	0.24	0.48	0.28	0.60	0.44	0.574
PAD (ours)	**0.88**	**0.84**	**0.80**	**0.80**	0.68	**0.92**	**0.72**	**0.96**	**0.88**	**0.725**
PAD w/o img	0.04	0.44	0.40	0.48	0.16	0.36	0.16	0.24	0.16	0.436
PAD w/o co-train	0.32	0.28	0.32	0.72	**0.92**	0.68	0.56	0.88	0.40	0.592

Table 1: Comparisons on Metaworld benchmark. We utilize a single policy to solve all 50 tasks in Metaworld. Due to the space limit, we show a subset of tasks and the average success rate on all 50 tasks. Detailed data can be found in Appendix A.4.

gripper status dimension. We collected 200 trajectories per task through teleoperation using a space mouse and scripted commands. Similarly, we developed a single policy capable of addressing all tasks, conditioned on instructions. We also assessed the policy's generalization capabilities on unseen tasks, as depicted in Figure 5.

Policy Training Details. As detailed in Section 3, the flexible PAD framework can be co-trained on various internet RGB video data and robotic demonstrations. In order to save computational resources and avoid the need to co-train the model from scratch in each robot domain, we first pre-train the model on internet data to establish better image prediction priors. We then adapt this pre-trained model to various robotic domains, including the simulated Metaworld and the real-world panda manipulation. Empirically, we first pretrain 200k steps on the BridgeData-v2 dataset [9], which consists of 60,000 trajectories. After this, we adapted the model to each domain, continuing training for an additional 100k steps with robotic demonstrations. The pre-training and adaptation stage requires approximately 2 days and 1 day, utilizing 4 NVIDIA A100 GPUs.

Moreover, we found that increasing the weight of the image prediction loss during the early adaptation stages accelerates convergence, as image priors are already established in the pre-trained models. Specifically, we maintained the image prediction loss coefficient λ_I at 1.0 throughout the training period and linearly increased λ_A and λ_E from 0.0 to 2.0 during the 100k training steps.

Policy Execution Details. Our policy is conditioned on the current image, c_I, and the robot pose, c_A, and predicts k frames of futures and actions. We configure the prediction horizon at $k = 3$ and set the interval between frames at $i = 4$ for both Metaworld and real-world tasks. During policy execution, we utilize 75 steps of DDIM sampling [5] to denoise the k steps of future images, $x_I^{1:K}$, and actions, $x_A^{1:k}$. These k step predictions can be viewed as k step planning and only the first predicted action, x_A^1, is executed by the robot. The robot then moves to the first desired pose using a simple linear interpolation motion planner, triggering the next prediction cycle.

Comparisons. Visual policy learning has been widely explored in previous studies. In our experiments, we opted to compare against a representative subset of prior methods that have either achieved state-of-the-art performance or share a similar architecture with our methods. Notably, all methods are trained on all tasks in the domain **using a single text-condition visual policy**.

- **Diffusion Policy [16].** A novel visual control policy that generates robot actions through an action diffuser. We augmented the original diffusion policy model with instruction conditions to address the multi-task setting. We use the CLIP encoder [25] as instruction encoders, referring to related work [29].

Task	Button-Press	Cable-Route	Pick (4 tasks)	Place (3 tasks)	Drawer-Open	Drawer-Close	Average
Diffusion Policy	0.70	0.30	0.28	0.34	0.42	0.58	0.38
SuSIE	0.74	0.44	0.46	0.40	0.42	0.70	0.49
RT-1	0.72	0.52	0.40	0.34	0.44	0.50	0.43
RT-2*	**0.80**	0.60	0.64	0.74	**0.66**	0.82	0.69
PAD(ours)	**0.80**	0.55	0.76	0.72	0.56	0.84	0.72
PAD-Depth(ours)	0.78	**0.64**	**0.84**	**0.78**	0.60	**0.88**	**0.78**

Table 2: Comparisons on real-world manipulation in-distribution tasks. PAD achieves the highest success rate. Incorporating depth modality can additionally lead to performance improvement. We evaluate each task with 50 roll-outs.
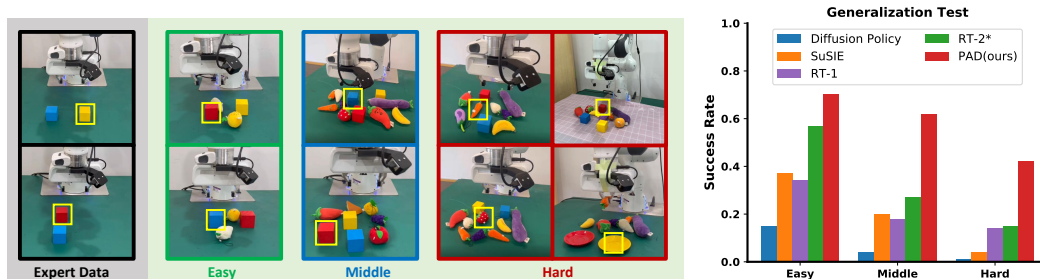

Figure 5: Generalization test under 3 levels of difficulties. The yellow bounding box suggests the target position. Our proposed PAD shows the strongest generalization abilities in unseen tasks.

- **SuSIE [18].** A two-stage approach that utilizes a pre-trained image-editing model [26] to generate image goals for robotic tasks, followed by a goal-conditioned low-level diffusion policy. We fine-tune the image-editing diffusion model on the same dataset and also use the diffusion policy for goal-conditioned behavioral cloning. To ensure a fair comparison, we also use the more powerful DiT framework as the image-editing model.

- **RT-1 [30].** An end-to-end robot control policy that leverages FiLM-conditioned [31] EfficientNet [32] to fuse visual input and language input, then followed by transformer blocks to output action.

- **RT-2* [33] (re-implement).** A large-scale embodied model that directly fine-tunes vision-language models(VLMs) to produce robot actions. The original RT-2 model was fine-tuned on the PaLM model [34], which is not publicly available. Following the specifications outlined in the original paper, we re-implemented the RT-2 model using the InstructBlip-7B [35] backbone.

- **GR-1 [36].** Method that also leverages image prediction to assist policy learning. Different from PAD, they generate images and actions via auto-regressive architecture.

4.2 Main Results

Performance Analysis. In all comparisons, we train a single visual policy to address all tasks within a domain, conditioned on instructions. Our proposed PAD outperforms all baselines by a significant margin. As shown in Table 1, in the Metaworld benchmark, PAD achieved an average success rate of 72.5%, which markedly surpasses the strongest baseline at 57.4%. Due to space constraints, we present comparisons on a subset of tasks and report the average success rate across all 50 tasks. A comprehensive comparison of all 50 tasks is available in Appendix A.4. Furthermore, Table 2 shows the results in real-world seen-tasks where PAD also attains the highest success rate.

We notice that PAD predicts more precise future images than the GR-1 method (Figure 6), likely due to the superior capabilities of diffusion models in image generation tasks. These precise images may more effectively facilitate policy learning, leading to higher success rates, particularly in tasks requiring precise and accurate operation such as picking small blocks, insertion, basketball, etc., in Metaworld.
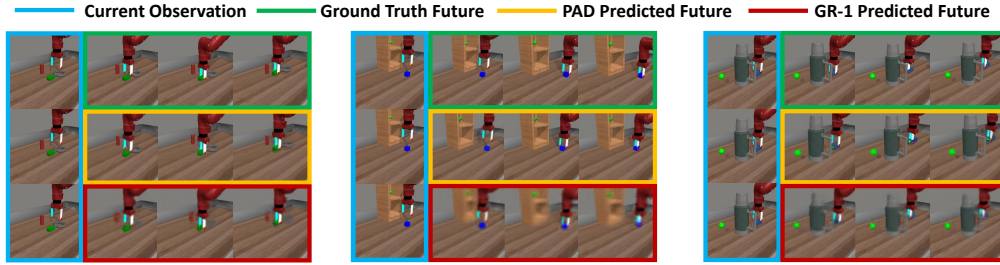
Figure 6: Comparisons on predicted images between PAD and GR-1. PAD generates more precise images than GR-1 which may potentially lead to more accurate control actions. Zoom in for better comparisons.

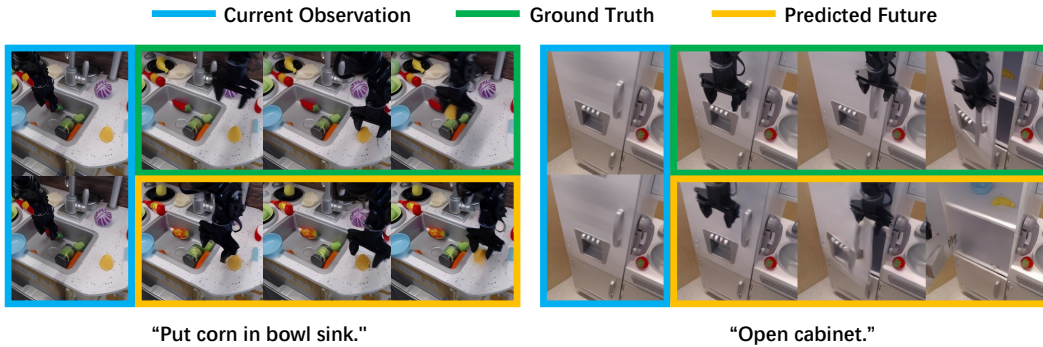"Put corn in bowl sink."   "Open cabinet."

Figure 7: Predictions on bridge datasets. PAD predicts futures align with instructions but also keeps uncertainty. In the first image, PAD imagines "a yellow pear" instead of the ground truth "banana"; in the second image, PAD imagines scenes faster than the ground truth.

Quality of the Generated Images. In addition to achieving the highest success rate in robotic control, we also present visualizations of some image prediction results in Figure 6 and Figure 7. In the Metaworld domain, the predicted image (second row) closely resembles the ground truth image (first row), which is directly decoded from the original latent. In the Bridge domain, the predicted image aligns with the language instructions but also keeps a certain level of uncertainty. These indicate that the PAD model has effectively learned the physical dynamics across these two domains.

4.3 Generalization Analysis

PAD can leverage existing physical knowledge from co-training on large-scale internet video datasets to enhance its generalization capabilities across new tasks. We evaluated PAD's generalization ability in real-world panda manipulation with unseen tasks. As depicted in Figure 4, the expert dataset comprises only colored square blocks and plates, while we introduce a variety of previously unseen fruit and vegetable toys during testing. We designed tasks of three difficulty levels: easy mode, featuring 1-4 disturbance objects; a middle level with 5-15 disturbance objects; and difficult tasks that require picking previously unseen objects with 5-15 disturbances or unseen backgrounds. We excluded depth input to ensure a fair comparison. As illustrated in Figure 5, PAD demonstrates remarkable generalization abilities, successfully managing out-of-distribution objects such as strawberries, carrots, and eggplants, and even adapting to new backgrounds. The baseline method failed to generalize to difficult unseen tasks.

4.4 Ablation Studies

Effectiveness of RGB image prediction. We evaluated the effectiveness of our joint prediction process by modifying the original model to **exclude** the image prediction component, namely in PAD w/o image prediction. This modification leads to significant performance drops compared to PAD, as illustrated in Table 1. The absence of image prediction compromises the robot's ability to utilize the physical knowledge encoded in the image modalities, which may be crucial for robotic control.
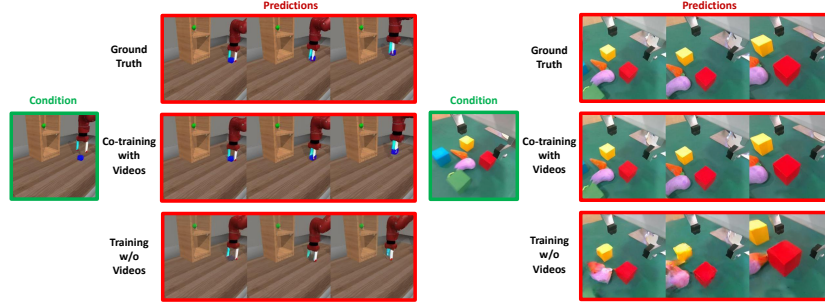
8

Figure 8: We observe that co-training with an internet video dataset leads to better image generation qualities, which may potentially lead to better robot action predictions.

Furthermore, predicting solely the robot pose provides only low-dimensional supervision signals, potentially leading to overfitting of the training data.

Effectiveness of Co-training with Internet RGB Video Datasets. Another major benefit of PAD is the ability to co-train with large-scale internet-sourced RGB videos, which potentially leverages the physical knowledge embedded in these web datasets to enhance robotic pose prediction. We train PAD without the inclusion of web-scale RGB data, namely PAD w/o co-train. We observed a performance drop without co-train on the video dataset, as shown in Table 1. Furthermore, the quality of the predicted image also decreased. For instance, as depicted in the bottom column of Figure 8, the blue block is absent in the predicted images. The quality of the predicted images markedly improves with co-training, which in turn indirectly enhances robot action prediction.

Compatible with Additional Modalities. As detailed in Section 3, our framework accommodates additional modalities owing to the adaptable DiT architectures. We incorporate additional depth image inputs in real-world manipulation experiments and jointly predict future RGB images, depth images, and robot actions, denoted as PAD-depth. We observe highly aligned prediction results among different modalities under our joint denoising framework, with some results illustrated in Figure 9. The inclusion of depth input enhances performance in manipulation tasks, as demonstrated in Table 2. This improvement may stem from the precise prediction of depth information, which aids agents in discerning distance changes, thereby enhancing performance. Moreover, our framework could be extended to predict other modalities relevant to robot control, such as tactile force or point clouds, which we left for the future work.
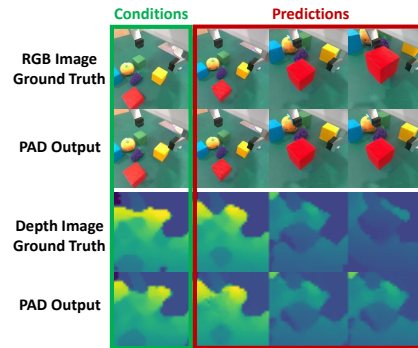
Figure 9: PAD can flexibly train with additional modality, and simultaneously predict all the futures through joint denoising process.

4.5 Scaling Analysis

We evaluated models across various sizes and patchify sizes [20], as outlined in Table 3. For example, the $XL/2$ model denotes the model with an XL size and a 2×2 patchify size. Halving the image patch size will quadruple the image token lengths, which leads to higher computational costs. Our findings reveal a strong correlation between computational allocation (measured as transformer Gflops) and the success rate (SR) of the learned policy, as depicted in Figure 10. All the experiments are run in Metaworld benchmarks and detailed success rates for each task are provided in Appendix A.5.

5 Related Work

Pre-training for Embodied Control. Vision-language pre-trained models, encoded with physical knowledge, can enhance embodied control from multiple aspects. Primarily, the pre-trained model can directly act as policy by either generating high-level plans [37, 38, 39, 40, 41] or producing direct

	PAD-XL/2	PAD-XL/4	PAD-XL/8	PAD-L/2	PAD-B/2
Layers	28	28	28	24	12
Hidden size	1152	1152	1152	1024	768
Heads	16	16	16	16	12
Token length	257	65	17	257	257
Parameters	661M	661M	661M	449M	128M
Gflops	119.1	29.5	7.7	79.1	22.5
Average SR	**72.5%**	64.5%	48.2%	68.4%	62.4%

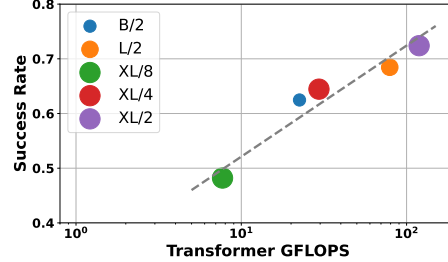Table 3: We test PAD performance under various sizes and computational costs.

Figure 10: Correlation between Transformer Gflops and policy success rate.

low-level motor control signals [30, 33, 42, 43, 44]. Many studies utilize the reasoning capabilities of pre-trained LLMs and VLMs to create high-level plans followed by motion primitives. Additionally, some approaches adapt pre-trained models to emit low-level motor control signals by adding an action head. Beyond directly acting as policy, pre-trained models can also guide policy learning from multiple aspects, such as providing good representations [45, 46, 47], providing reward signals [48, 49, 50], synthesizing goal images [18, 51], and predicting future sequences [17].

Diffusion Models for Embodied Control. Recently, diffusion models have been adopted to tackle challenges in embodied control. A subset of research focuses on training conditional diffusion models that guide behavior synthesis based on desired rewards, and constraints under low dimensional state-input setting [52, 53, 54, 55]. Diffusion Policy [16] trains a visual-motor policy to be conditioned on RGB observations and can better express the multimodal action distributions. However, these methods develop task-specific policies from scratch, missing out on the benefits of pre-training with internet data. Another strand of research utilizes large-scale pre-trained diffusion models to perform data augmentation on training data, such as GenAug [56], ROSIE [57], and CACTI [58].

Future Prediction for Policy Learning. There also exist works that leverage future image predictions to assist policy learning. GR-1 [36] employs an autoregressive transformer to sequentially predict future images and actions. In contrast, we adopt a joint diffusion architecture that predicts more accurate future images, potentially leading to improved policy learning performance. UniPi[17] and SuSIE [18] employ a two-stage policy learning process, initially using a diffusion generative model to forecast future image or video sequences, and subsequently training an inverse dynamics model or a low-level diffusion policy based on these goal images. In contrast to these two-stage methods, our approach presents distinct advantages. First, while previous methods utilize diffusion models with a CNN-based U-net backbone [23], designed primarily for image generation and limited to visual predictions, our method adopts a diffusion transformer (DiT) architecture [20]. This architecture adeptly handles multiple modalities concurrently via straightforward token concatenation and attention-mask mechanisms, enabling us to jointly predict future and actions simultaneously. Secondly, using images as the interface between prediction and action may not fully leverage the encoded features inside pre-trained diffusion models. The effectiveness of these two-stage methods depends heavily on the quality of the generated images. In contrast, our model integrates image generation and robotic action within a unified denoising process.

6 Conclusion and Discussion

We present PAD, a novel framework to predict future images and generate actions under a joint denoising process. Moreover, PAD can co-train with internet video datasets and extend to other robotic modalities. Both simulated and real-world experiments demonstrated the efficiency of PAD.

A limitation of the current method is that we only tested with three types of modalities. Subsequent endeavors could extend this framework to incorporate additional robot-related input data, such as tactile information, which we believe are valuable research directions. Another limitation is that the control frequency of PAD is not very high since we need to jointly denoise the images and actions. Future work can explore efficient ways to leverage image predictions, such as utilizing the intermediate latent space of predicted images rather than the high-dimensional pixel spaces.

References

[1] Sarah-Jayne Blakemore and Chris Frith. The role of motor contagion in the prediction of action. *Neuropsychologia*, 43(2):260–267, 2005.

[2] Andreja Bubic, D Yves Von Cramon, and Ricarda I Schubotz. Prediction, cognition and the brain. *Frontiers in human neuroscience*, 4:1094, 2010.

[3] James Moore and Patrick Haggard. Awareness of action: Inference and prediction. *Consciousness and cognition*, 17(1):136–144, 2008.

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

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

[6] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[8] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017.

[9] Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pages 1723–1736. PMLR, 2023.

[10] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

[11] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.

[12] Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6007–6017, 2023.

[13] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022.

[14] Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *arXiv preprint arXiv:2205.15868*, 2022.

[15] Xin Ma, Yaohui Wang, Gengyun Jia, Xinyuan Chen, Ziwei Liu, Yuan-Fang Li, Cunjian Chen, and Yu Qiao. Latte: Latent diffusion transformer for video generation. *arXiv preprint arXiv:2401.03048*, 2024.

[16] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

[17] Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *Advances in Neural Information Processing Systems*, 36, 2024.

[18] Kevin Black, Mitsuhiko Nakamoto, Pranav Atreya, Homer Walke, Chelsea Finn, Aviral Kumar, and Sergey Levine. Zero-shot robotic manipulation with pretrained image-editing diffusion models. *arXiv preprint arXiv:2310.10639*, 2023.

[19] Anurag Ajay, Seungwook Han, Yilun Du, Shuang Li, Abhi Gupta, Tommi Jaakkola, Josh Tenenbaum, Leslie Kaelbling, Akash Srivastava, and Pulkit Agrawal. Compositional foundation models for hierarchical planning. *Advances in Neural Information Processing Systems*, 36, 2024.

[20] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

[21] 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.

[22] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022.

[23] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn 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.

[24] Martin Riedmiller and A Lernen. Multi layer perceptron. *Machine Learning Lab Special Lecture, University of Freiburg*, 24, 2014.

[25] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[26] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18392–18402, 2023.

[27] Jianlan Luo, Zheyuan Hu, Charles Xu, You Liang Tan, Jacob Berg, Archit Sharma, Stefan Schaal, Chelsea Finn, Abhishek Gupta, and Sergey Levine. Serl: A software suite for sample-efficient robotic reinforcement learning. *arXiv preprint arXiv:2401.16013*, 2024.

[28] Jianlan Luo, Charles Xu, Fangchen Liu, Liam Tan, Zipeng Lin, Jeffrey Wu, Pieter Abbeel, and Sergey Levine. Fmb: a functional manipulation benchmark for generalizable robotic learning. *arXiv preprint arXiv:2401.08553*, 2024.

[29] Huy Ha, Pete Florence, and Shuran Song. Scaling up and distilling down: Language-guided robot skill acquisition, 2023.

[30] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[31] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[32] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[33] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[34] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

[35] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. *Advances in Neural Information Processing Systems*, 36, 2024.

[36] Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. *arXiv preprint arXiv:2312.13139*, 2023.

[37] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[38] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. *arXiv preprint arXiv:2209.07753*, 2022.

[39] Andy Zeng, Adrian Wong, Stefan Welker, Krzysztof Choromanski, Federico Tombari, Aveek Purohit, Michael Ryoo, Vikas Sindhwani, Johnny Lee, Vincent Vanhoucke, et al. Socratic models: Composing zero-shot multimodal reasoning with language. *arXiv preprint arXiv:2204.00598*, 2022.

[40] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022.

[41] Yanjiang Guo, Yen-Jen Wang, Lihan Zha, Zheyuan Jiang, and Jianyu Chen. Doremi: Grounding language model by detecting and recovering from plan-execution misalignment. *arXiv preprint arXiv:2307.00329*, 2023.

[42] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022.

[43] Jianke Zhang, Yanjiang Guo, Xiaoyu Chen, Yen-Jen Wang, Yucheng Hu, Chengming Shi, and Jianyu Chen. Hirt: Enhancing robotic control with hierarchical robot transformers. In *8th Annual Conference on Robot Learning*.

[44] Yen-Jen Wang, Bike Zhang, Jianyu Chen, and Koushil Sreenath. Prompt a robot to walk with large language models. *arXiv preprint arXiv:2309.09969*, 2023.

[45] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[46] Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022.

[47] Arjun Majumdar, Karmesh Yadav, Sergio Arnaud, Jason Ma, Claire Chen, Sneha Silwal, Aryan Jain, Vincent-Pierre Berges, Tingfan Wu, Jay Vakil, et al. Where are we in the search for an artificial visual cortex for embodied intelligence? *Advances in Neural Information Processing Systems*, 36, 2024.

[48] Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. *arXiv preprint arXiv:2310.12931*, 2023.

[49] Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge. *Advances in Neural Information Processing Systems*, 35:18343–18362, 2022.

[50] Ademi Adeniji, Amber Xie, Carmelo Sferrazza, Younggyo Seo, Stephen James, and Pieter Abbeel. Language reward modulation for pretraining reinforcement learning. *arXiv preprint arXiv:2308.12270*, 2023.

[51] Ivan Kapelyukh, Vitalis Vosylius, and Edward Johns. Dall-e-bot: Introducing web-scale diffusion models to robotics. *IEEE Robotics and Automation Letters*, 2023.

[52] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

[53] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.

[54] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

[55] Julen Urain, Niklas Funk, Jan Peters, and Georgia Chalvatzaki. Se (3)-diffusionfields: Learning smooth cost functions for joint grasp and motion optimization through diffusion. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5923–5930. IEEE, 2023.

[56] Zoey Chen, Sho Kiami, Abhishek Gupta, and Vikash Kumar. Genaug: Retargeting behaviors to unseen situations via generative augmentation. *arXiv preprint arXiv:2302.06671*, 2023.

[57] Tianhe Yu, Ted Xiao, Austin Stone, Jonathan Tompson, Anthony Brohan, Su Wang, Jaspiar Singh, Clayton Tan, Jodilyn Peralta, Brian Ichter, et al. Scaling robot learning with semantically imagined experience. *arXiv preprint arXiv:2302.11550*, 2023.

[58] Zhao Mandi, Homanga Bharadhwaj, Vincent Moens, Shuran Song, Aravind Rajeswaran, and Vikash Kumar. Cacti: A framework for scalable multi-task multi-scene visual imitation learning. *arXiv preprint arXiv:2212.05711*, 2022.

A Appendix

Videos of PAD can be found at `https://sites.google.com/view/pad-paper`.

A.1 Additional Implementation Details of PAD

A.1.1 Input Encoder and Output Decoders

Image Encoder and Tokenizer. The image encoder is a frozen VAE encoder same as [20]. Take PAD-XL/2 model for example, the encoded latent space for 256×256 image is $32 \times 32 \times 4$, and patchify into $(32/2) * (32/2) = 256$ patches, which then are tokenized into 256 tokens.

Robot action Encoder and Tokenizer. The robot action is concatenated into a vector and passed into MLP layers, we predict k steps of the future each with 7-dimensional poses, which totally consists $(k + 1) * 7$ dimensional vectors ($(k + 1) * 4$ in Metaworld), and then this vector is tokenized into 1 token.

Depth image Encoder and Tokenizer (If presented). We directly down-sample the depth image to a size of $32 * 32 * 1$, and follow the same patchfy process as the RGB image. The patch size is set to 8. The patchfy for depth image resulted in $(32/8) * (32/8) = 16$ patches, which then are tokenized into 16 tokens.

Output Decoder. The decoder part mainly inverses the encoder part. The decoder process first reconstructs the future latent from the token output by DiT, then adopts the corresponding decoder to recover the original samples in each modality.

A.2 Additional Implementation Details of Baselines

The RT-1 baseline is based on official implementation `https://github.com/google-research/robotics_transformer`.

The Diffusion Policy baseline is based on `https://github.com/real-stanford/diffusion_policy`, and we follow `https://github.com/real-stanford/scalingup` to add language condition.

The RT-2 baseline is re-implemented by ourselves. We use the InstructBlip-vicuna-7b model as backbone `https://huggingface.co/Salesforce/instructblip-vicuna-7b`.

The GR-1 baseline is built on `https://github.com/bytedance/GR-1`. Since we can not access the pretraining dataset in the original paper, we initialize the model with the author's open-source checkpoint.

A.2.1 Additional Model Training Details

	PAD-XL/2	PAD-XL/4	PAD-XL/8	PAD-L/2	PAD-B/2
Layers	28	28	28	24	12
Hidden size	1152	1152	1152	1024	768
Heads	16	16	16	16	12
Parameters	661M	661M	661M	449M	128M
Gflops	119.1	29.5	7.7	79.1	22.5
Learning Rate	1e-4	1e-4	1e-4	1e-4	1e-4
Batch size	256	256	256	256	256
Input image shape	256*256	256*256	256*256	256*256	256*256
Input noised latent	32*32*(4*4)	32*32*(4*4)	32*32*(4*4)	32*32*(4*4)	32*32*(4*4)
Patchify size	2*2	4*4	8*8	2*2	2*2
Image token size	256	64	16	256	256
Input robot action shape	1*28	1*28	1*28	1*28	1*28
Action token size	1	1	1	1	1
Total token size	257	65	17	257	257

Table 4: Models with various size and computational cost.

A.3 Real world Experiment Details

Expert data collection. We collected data on 6 categories of tasks, including button press, cable-route, pick and place, and drawer open/close. For the pick task, we include 4 colors of blocks. For the place task, we include 3 colors of plate. We randomly placed 1-5 objects on the table and asked the robot to pick/place certain objects conditioned on instruction. Some samples of expert demonstrations are visualized in Figure 11.

Generalization Test Task Samples. We test the generalization ability of learned policy under numerous unseen objects. The unseen task is much more complicated than expert tasks, as shown in Figure 12. For convenience, videos of PAD can be found at `https://sites.google.com/view/pad-paper`.
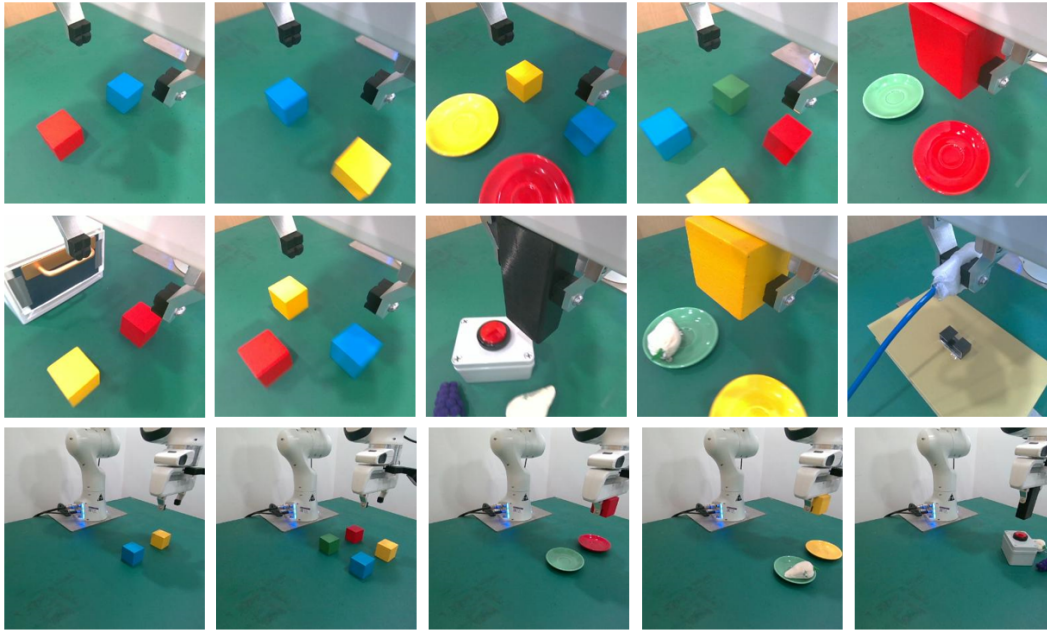

Figure 11: Samples of tasks that we collected demonstrations.

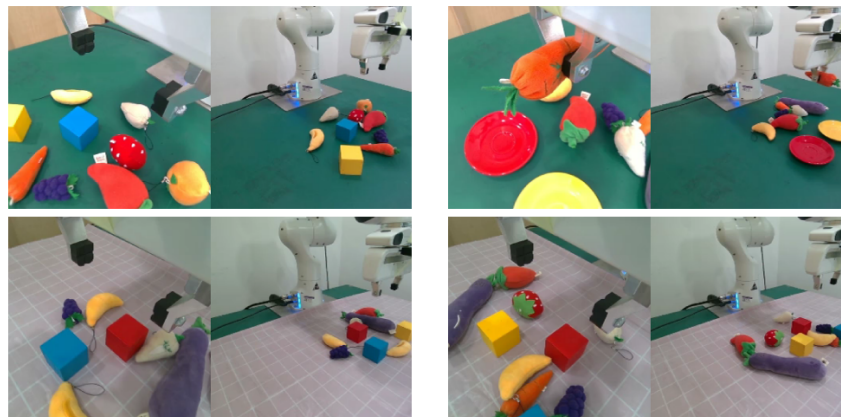Figure 12: Samples of **unseen** tasks used for generalization test.

A.4   Details Baselines and Ablations in Metaworld

Task	Diffusion Policy	SuSIE	RT-1	RT-2*	GR-1	PAD	PAD w/o img	PAD w/o cotrain
assembly-v2	0.20	0.40	0.00	0.24	0.64	**0.88**	0.04	0.32
basketball-v2	0.08	0.24	0.00	0.08	0.08	**0.84**	0.44	0.28
button-press-topdown-v2	0.16	0.32	**1.00**	0.84	**1.00**	0.92	0.92	0.92
button-press-top-wall-v2	0.00	0.24	0.60	0.88	0.80	**0.96**	0.64	0.84
button-press-v2	0.92	0.96	0.88	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
button-press-wall-v2	0.44	0.60	**1.00**	0.44	0.48	0.68	0.48	0.60
coffee-button-v2	0.64	0.80	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
coffee-pull-v2	0.00	0.32	0.08	0.68	0.52	**0.80**	0.40	0.32
coffee-push-v2	0.16	0.32	0.04	**0.76**	0.44	0.52	0.48	0.64
dial-turn-v2	0.52	0.52	0.00	0.52	0.44	**0.56**	0.28	0.52
disassemble-v2	0.00	0.72	0.00	0.12	0.40	**0.88**	0.24	0.64
door-close-v2	0.52	0.56	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.96
door-open-v2	0.32	0.68	0.56	0.96	**1.00**	**1.00**	0.88	0.92
drawer-close-v2	0.52	0.88	**1.00**	0.96	0.96	**1.00**	0.88	0.96
drawer-open-v2	0.36	0.60	0.56	0.92	**1.00**	**1.00**	**1.00**	**1.00**
faucet-open-v2	0.64	0.36	0.88	**1.00**	**1.00**	**1.00**	0.40	**1.00**
faucet-close-v2	0.76	0.56	**1.00**	0.96	0.96	**0.92**	0.92	0.92
hammer-v2	0.08	0.04	0.00	0.20	0.48	**0.80**	0.48	0.72
handle-press-side-v2	0.44	0.72	**1.00**	0.76	0.48	0.40	0.76	0.40
handle-press-v2	0.72	0.68	**1.00**	0.96	0.80	0.80	0.96	0.80
lever-pull-v2	0.20	0.32	0.32	0.20	**0.4**	0.32	0.28	0.08
peg-insert-side-v2	0.16	0.24	0.00	0.12	0.24	0.68	0.16	**0.92**
peg-unplug-side-v2	0.28	0.20	0.32	0.48	0.32	**0.60**	0.16	0.20
pick-out-of-hole-v2	0.00	0.16	0.00	0.24	0.24	**0.52**	0.00	0.36
pick-place-wall-v2	0.36	0.24	0.00	0.32	0.48	**0.92**	0.36	0.68
pick-place-v2	0.08	0.24	0.00	0.48	0.72	**0.80**	0.20	0.20
plate-slide-v2	0.60	0.68	0.08	**0.88**	0.88	0.72	0.16	0.48
plate-slide-side-v2	**1.00**	**1.00**	0.84	**1.00**	**1.00**	0.88	0.36	0.68
plate-slide-back-v2	0.00	0.80	0.20	0.72	0.36	**1.00**	0.36	0.04
plate-slide-back-side-v2	0.00	**0.16**	0.00	0.12	0.00	0.12	0.00	0.12
soccer-v2	0.00	0.08	0.28	0.28	0.12	0.08	0.08	**0.32**
stick-push-v2	0.12	0.20	0.00	0.12	0.60	**0.96**	0.24	0.88
stick-pull-v2	0.00	0.16	0.00	0.00	0.44	**0.88**	0.16	0.40
push-wall-v2	0.08	0.16	0.00	0.68	0.32	**0.84**	0.20	0.44
push-v2	0.20	0.32	0.00	0.40	0.44	0.52	0.16	**0.60**
reach-wall-v2	0.72	0.92	0.12	0.76	**1.00**	**1.00**	0.92	0.92
reach-v2	0.40	0.60	0.76	0.40	**1.00**	**1.00**	0.44	**1.00**
shelf-place-v2	0.00	0.08	0.00	0.20	0.28	**0.72**	0.16	0.56
sweep-into-v2	0.08	0.16	0.12	0.36	0.16	**0.40**	0.24	0.28
sweep-v2	0.04	0.16	0.24	0.60	0.72	**0.80**	0.20	0.44
window-open-v2	0.60	0.68	**1.00**	**1.00**	**1.00**	0.92	**1.00**	0.96
window-close-v2	0.36	0.96	**1.00**	0.96	**1.00**	**1.00**	**1.00**	0.96
bin-picking-v2	0.00	0.24	0.00	0.12	0.40	**0.88**	0.00	0.72
box-close-v2	0.00	0.20	0.00	0.00	0.40	**0.52**	0.16	0.60
door-lock-v2	0.12	0.32	0.00	0.40	0.60	**0.88**	0.12	0.72
door-unlock-v2	0.44	0.44	0.28	0.52	0.44	**0.60**	0.68	0.56
hand-insert-v2	0.08	0.24	0.16	0.28	0.20	**0.40**	0.08	0.20
push-back-v2	0.00	0.04	0.00	0.20	**0.52**	0.28	0.00	**0.52**
Average	27.9%	41.0%	34.6%	52.2%	57.4%	**72.5%**	43.6%	59.2%

Table 5: Detailed success rate of baselines and ablations. We did not include the handle-pull-side-v2 and handle-pull-v2 tasks since the expert policy for these two tasks in the original benchmark had low success rates. Every task is tested with 25 rollouts.

A.5 Detailed Scaling Results

Task	XL/8	B/2	L/2	XL/4	XL/2
assembly-v2	0.32	**0.96**	**0.96**	0.20	0.88
basketball-v2	0.00	0.36	0.32	0.40	**0.84**
button-press-topdown-v2	**1.00**	0.76	**1.00**	**1.00**	0.92
button-press-topdown-wall-v2	0.72	**1.00**	0.90	0.90	0.96
button-press-v2	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
button-press-wall-v2	**0.88**	0.56	0.60	0.40	0.68
coffee-button-v2	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
coffee-pull-v2	0.32	0.10	0.30	0.68	**0.80**
coffee-push-v2	0.28	0.60	0.40	0.52	**0.52**
dial-turn-v2	0.28	0.40	**0.56**	0.20	0.56
disassemble-v2	0.52	0.80	0.92	**1.00**	0.88
door-close-v2	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
door-open-v2	0.92	**1.00**	**1.00**	**1.00**	**1.00**
drawer-close-v2	0.52	**1.00**	**1.00**	**1.00**	**1.00**
drawer-open-v2	0.88	**1.00**	**1.00**	**1.00**	**1.00**
faucet-open-v2	0.32	**1.00**	**1.00**	**1.00**	**1.00**
faucet-close-v2	0.92	0.44	0.96	**1.00**	0.92
hammer-v2	0.60	**1.00**	0.68	0.92	0.80
handle-press-side-v2	0.40	0.48	**0.72**	0.60	0.40
handle-press-v2	0.48	0.92	0.84	0.80	**0.96**
lever-pull-v2	0.00	0.20	0.64	0.12	**0.32**
peg-insert-side-v2	**0.80**	0.52	**0.80**	**0.80**	0.68
peg-unplug-side-v2	0.44	0.32	0.56	0.48	**0.60**
pick-out-of-hole-v2	0.36	0.10	0.16	0.50	**0.52**
pick-place-wall-v2	0.20	0.68	0.56	0.60	**0.92**
pick-place-v2	0.32	0.32	0.76	0.52	**0.80**
plate-slide-v2	0.52	**0.76**	0.60	0.48	0.72
plate-slide-side-v2	0.40	**0.96**	0.52	0.92	0.88
plate-slide-back-v2	0.20	0.36	0.12	0.20	**1.00**
plate-slide-back-side-v2	0.04	**0.20**	0.04	0.12	0.12
soccer-v2	0.12	0.00	0.32	0.24	**0.08**
stick-push-v2	**1.00**	**1.00**	**1.00**	**1.00**	0.96
stick-pull-v2	0.32	**0.92**	0.64	0.40	0.88
push-wall-v2	0.60	0.40	**0.84**	0.80	**0.84**
push-v2	0.28	0.32	0.92	0.64	**0.52**
reach-wall-v2	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
reach-v2	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
shelf-place-v2	0.40	0.72	**0.90**	0.80	0.72
sweep-into-v2	0.12	0.40	0.48	0.36	**0.40**
sweep-v2	0.48	0.64	**0.88**	0.80	0.80
window-open-v2	0.32	**1.00**	**1.00**	0.90	0.92
window-close-v2	0.28	**1.00**	**1.00**	0.72	**1.00**
bin-picking-v2	0.88	0.72	0.80	**1.00**	0.88
box-close-v2	0.40	0.40	**0.72**	0.52	0.52
door-lock-v2	0.60	**0.88**	0.76	0.84	**0.88**
door-unlock-v2	0.40	**0.68**	0.30	0.64	0.60
hand-insert-v2	0.12	0.00	0.16	0.20	**0.40**
push-back-v2	0.16	0.20	**0.52**	0.12	0.28
handle-pull-side-v2	N/A	N/A	N/A	N/A	N/A
handle-pull-v2	N/A	N/A	N/A	N/A	N/A
Average	48.2%	62.4%	68.4%	64.5%	**72.5%**

Table 6: Detailed success rate under different model sizes and computational allocations. We did not include the handle-pull tasks since the expert policy for these two tasks are in low success rates.

A.6 Instructions used in tasks

Task	Instructions
assembly-v2	assemble the object
basketball-v2	shoot the basketball
button-press-topdown-v2	press the button
button-press-topdown-wall-v2	press the button
button-press-v2	press the button
button-press-wall-v2	press the button
coffee-button-v2	press the button
coffee-pull-v2	pull back cup
coffee-push-v2	push forward cup
dial-turn-v2	turn the dial
disassemble-v2	disassemble the object
door-close-v2	close the door
door-open-v2	open the door
drawer-close-v2	close the drawer
drawer-open-v2	open the drawer
faucet-open-v2	open the faucet
faucet-close-v2	close the faucet
hammer-v2	pick up hammer
handle-press-side-v2	press the handle
handle-press-v2	press the handle
lever-pull-v2	pull the lever
peg-insert-side-v2	insert the peg
peg-unplug-side-v2	unplug the peg
pick-out-of-hole-v2	pick red object
pick-place-wall-v2	pick red object
pick-place-v2	pick red object
plate-slide-v2	slide forward plate
plate-slide-side-v2	slide side plate
plate-slide-back-v2	slide back plate
plate-slide-back-side-v2	slide back plate
soccer-v2	kick the soccer
stick-push-v2	push the stick
stick-pull-v2	pull the stick
push-wall-v2	push the object
push-v2	pick red object
reach-wall-v2	reach red object
reach-v2	reach red object
shelf-place-v2	place blue object
sweep-into-v2	sweep brown box
sweep-v2	sweep brown box
window-open-v2	open the window
window-close-v2	close the window
bin-picking-v2	pick green object
box-close-v2	close the box
door-lock-v2	lock the door
door-unlock-v2	unlock the door
hand-insert-v2	put box into hole
handle-pull-side-v2	pull the handle
handle-pull-v2	pull the handle
push-back-v2	push back object

Table 7: Instructions for each tasks in metaworld. We mainly designed the instruction based on the task names.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We carefully claim our contribution under specific settings.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We include the limitation part in the last section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: The paper does not include theoretical results

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We carefully describe the method details in the paper.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Code is included in supplementary material.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: we specify details in the Experiment section.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We provide detailed data in the Appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We describe the computational cost in Experiment section.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

Answer: [Yes]

Justification: The research conducted in the paper conforms, in every respect, with the NeurIPS Code of Ethics

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: We did not see any negative social impact at this moment.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper poses no such risks

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We cite the original paper that produced the code package or dataset

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.