Diffusion Transformer Policy

Zhi Hou[1]* Tianyi Zhang[2,1]* Yuwen Xiong[1] Hengjun Pu[3,1] Chengyang Zhao[4,1]
Ronglei Tong[5] Yu Qiao[1] Jifeng Dai[6,1] Yuntao Chen[7]†
[1] Shanghai AI Lab [2] College of Computer Science and Technology, Zhejiang University
[3] MMLab, The Chinese University of Hong Kong [4] Peking University [5] SenseTime Research
[6] Tsinghua University [7] Center for Artificial Intelligence and Robotics, HKISI, CAS

Abstract

Recent large vision-language-action models pretrained on diverse robot datasets have demonstrated the potential for generalizing to new environments with a few in-domain data. However, those approaches usually predict individual discretized or continuous action by a small action head, which limits the ability in handling diverse action spaces. In contrast, we model the continuous action sequence with a large multi-modal diffusion transformer, dubbed as Diffusion Transformer Policy, in which we directly denoise action chunks by a large transformer model rather than a small action head for action embedding. By leveraging the scaling capability of transformers, the proposed approach can effectively model continuous end-effector actions across large diverse robot datasets, and achieve better generalization performance. Extensive experiments demonstrate the effectiveness and generalization of Diffusion Transformer Policy on Maniskill2, Libero, Calvin and SimplerEnv, as well as the real-world Franka arm, achieving consistent better performance on Real-to-Sim benchmark SimplerEnv, real-world Franka Arm and Libero compared to OpenVLA and Octo. Specifically, without bells and whistles, the proposed approach achieves state-of-the-art performance with only a single third-view camera stream in the Calvin task ABC→D, improving the average number of tasks completed in a row of 5 to 3.6, and the pretraining stage significantly facilitates the success sequence length on the Calvin by over 1.2. Project Page: `https://zhihou7.github.io/dit_policy_vla/`

1. Introduction

Traditional robot learning paradigm usually relies on large-scale data collected for a specific robot and task, but collecting robot data for generalist tasks is time-consuming and expensive due to the limitations of robot hardware in the real world. Nowadays, the foundational models [37, 46–48, 58] in Natural Language Process and Computer Vision, pretrained on broad, diverse, task-agnostic datasets, have demonstrated powerful ability in solving downstream tasks either zero-shot or with a few task-specific samples. It is principally possible that a general robot policy exposed to large scale diverse robot datasets improves generalization and performance on downstream tasks [6, 7]. However, it is challenging to train a general robot policy on a large scale of cross-embodiment datasets with diverse sensors, action spaces, tasks, camera views, and environments.

Toward a unified robot policy, existing works directly map visual observation and language instructions to actions with large vision-language-action models for robot navigation [62, 63] or manipulation [6, 7, 28, 69], and demon-

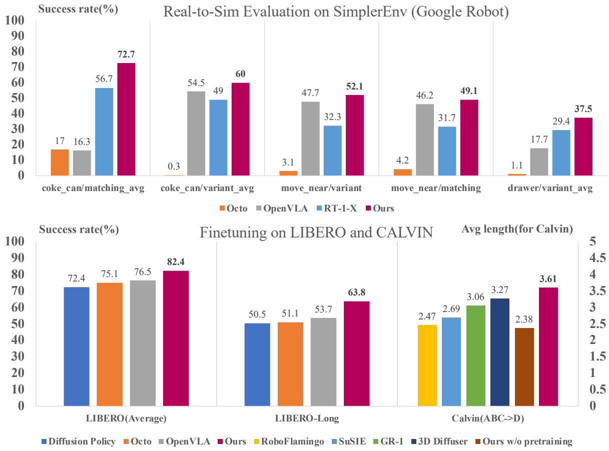


Figure 1. Comparisons between previous state-of-the-art approaches and ours on popular generalist vision-language-action simulation benchmarks. LIBERO [36] and Calvin [43] are for finetuning generalization, while SimplerEnv [34] is to evaluate the generalization to environment variances under Real-to-Sim Benchmark.

*Equal Contribution

†Corresponding Author

strate zero-shot or few-shot generalization to new environments. Robot Transformers [6, 7, 50] present robot policy based on transformer architecture, and demonstrate robust generalization by training on the large scale of Open X-Embodiment Dataset [50]. Octo [69] follows the autoregressive transformer architecture with a diffusion action head, while OpenVLA [28] discretizes the action space and leverage the pretrained vision-language model to build VLA model exposed to Open X-Embodiment Dataset [50]. Though those Vision-Language-Action (VLA) models [28, 69] have shown the potential to learn robot policy from the large cross embodiment datasets [50], the diversity of robot space among the cross embodiment datasets still limits the generalization.

Recent diffusion policy [12, 26, 57, 77] has shown its stable ability in robot policy learning for single task imitation learning with UNet or cross attention architecture, and diffusion transformer demonstrates its scalability in multi-modal image generation [52]. Specifically, Octo [69] presents a generalist policy that denoises the action with a small MLP network conditioned on a single embedding of auto-regressive multi-modal transformer. However, the robot space of large-scale cross-embodiment datasets contains various cameras views and diverse action spaces, which poses a significant challenge for a small MLP to separately denoise each continuous action conditioned on a single action head embedding. Meanwhile, previous diffusion policies [26, 57, 69] first fuse historical image observations and instruction into embeddings before the denoising process, which might limit action denoising learning since action anticipation usually relies directly on detailed historical observations rather than fused embedding.

In this paper, we design a Diffusion Transformer architecture for generalist robot policy learning. Similar to previous robot transformer models [6, 7, 28, 50, 69], we leverage the transformer as our base module to retain the scalability on the large-scale cross-embodiment datasets. Different from [6, 7, 28, 50, 69], we present an in-context conditional diffusion transformer architecture to denoise the action chunks (*i.e.*, action sequence), rather than utilizing a small shared MLP to separately denoise each action embedding to continuous actions as illustrated in Figure 2. Meanwhile, the in-context conditional design enables the action denoising learning directly condition on each image observation patches, which supports the denoising model in perceiving the subtle nuances (*e.g.*, action delta) in historical visual observations. Also, DiT Policy retains the scalability of transformer for diffusion, allowing for more effective generalization from large and diverse embodiment datasets.

In a nutshell, we present a Diffusion Transformer Policy, that incorporates a causal transformer as an in-context conditional diffusion backbone and denoise continuous action chunks. Extensive experiments demonstrate DiT Policy achieves considerably better performance on two large-scale Sim datasets, Maniskill2 (novel camera views) and Calvin, compared to baselines. Meanwhile, the proposed model trained on the Open X-Embodiment Dataset achieves better generalization performance compared to Octo and OpenVLA on the Real Franka and Libero.

2. Related Work

Diffusion Policy Denoise diffusion techniques [8, 15, 21, 52, 59] are pioneering image generation, and recent Diffusion Policy [9, 10, 12, 26, 35, 57, 71, 72, 77] has exhibited a powerful ability in modeling multimodal actions compared to previous robot policy strategies. Current diffusion policy approaches usually follow an Unet structure or a shallow cross-attention network for a single manipulation task, leaving large-scale multimodal diffusion policy poorly investigated. For example, 3D diffusion Policy [77] denoises policy conditioned on a 3D point cloud, while 3D diffuser actor [26] proposes a 3D diffusion strategy based on point cloud with cross-attention. Differently, we present a scalable in-context conditioning diffusion transformer architecture, which directly conditions on each historical observation. Recent generalist policy Octo [69] conditions the denoise process on the embedding from the Transformer model with a small MLP diffuser. By contrast, the diffuser in Diffusion Transformer Policy is a large Transformer architecture.

Generalist Robot Policies Language-conditioned policy [11, 19, 39, 45, 56, 78] is more suitable and general for real applications, and the embodiment community has shown increasing interest in generalist robot policy with foundational multi-modal models for both robot navigation [4, 23, 62, 63, 67, 76] and manipulation [2, 5–7, 16, 17, 25, 28, 40, 42, 50, 51, 56, 60, 64, 65, 69, 74, 75]. Recent approaches [6, 7, 28, 50, 69] aim to achieve generalist policy with scalable Vision-Language-Action models. We follow this paradigm to approach generalist and adaptive robot policy. [6, 7, 28, 50] construct the action token by discretizing each dimension of the robot actions separately into 256 bins. However, this discretization strategy incurs internal deviation in robot execution. Unlike those methods, we present a Diffusion Transformer Generalist Policy, which denoises the continuous actions with a large Transformer model. The proposed approach retains the scalability of the Transformer and meanwhile facilitates the modeling of cross-embodiment action chunk representations. Meanwhile, the Diffusion Transformer Policy aligns robot action together with the language instructions and image observations as an in-context conditional style.

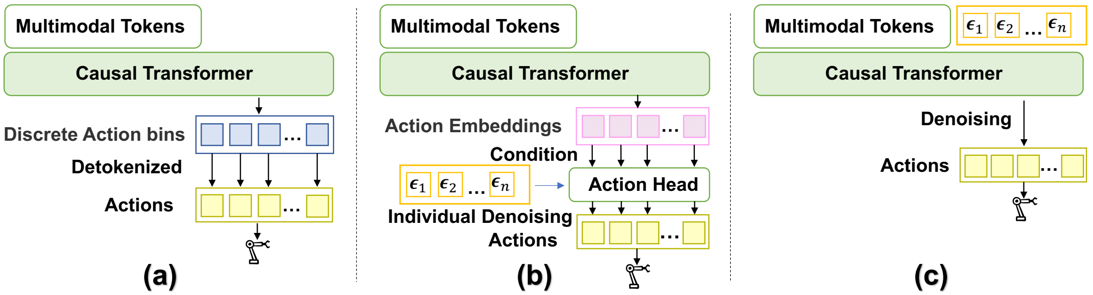


Figure 2. Illustrations of different robot policy architectures. (a) is the common robot transformer architecture with discretization actions, *e.g*., Robot Transformer [6, 7] and OpenVLA [28]. (b) is the transformer architecture with diffusion action head which denoises the individual continuous action with a small network condition on each embedding from the causal transformer, *e.g*., Octo [69]. (c) is the proposed Diffusion Transformer architecture that utilizes the large transformer to denoise actions in an in-context conditioning style.

3. Method

We describe the architecture of diffusion transformer policy in this section, a DiT-based generalist diffusion policy that can be adapted to new environments and embodiments.

3.1. Architecture

Instruction Tokenization. The language instructions are tokenized by a frozon CLIP [55] model.

Image observation Representations. The image observations first pass into the DINOv2 [49] to obtain the image patch features. Note that DINOv2 is trained on the web data which is different from the robot data, we thus jointly optimize the DINOv2 parameters together with Transformers through an end-to-end way.

Q-Former. To reduce the computation cost, a Q-Former [30] together with FiLM [53] conditioning is incorporated to select image features from the patch features of DINOv2 [49] by instruction context.

Action Preprocess. We use the end-effector action and represent each action with a 7D vector, including 3 dimensions for the translation vector, 3 dimensions for the rotation vector, and a dimension for the gripper position. To align the dimension with image and language tokens, we simply pad the continuous action vector with zeros to construct the action representation. We only add the noise into the 7D action vector during denoise diffusion optimization.

Architecture. Our core design is the Diffusion Transformer structure [52] which denoises action token chunks, instead of each single action token, conditioned directly on image observation and instruction tokens by an in-context conditioning style with a causal transformer, *i.e*., we simply concatenate image features, language tokens, and timestep embedding in the front of the sequence, equally treating the noisy action from the instruction tokens as illustrated in Figure 3. This design retains the scaling properties of transformer networks, and allows the denoising learning conditioned directly on image patches, thus facilitating the model capture the detailed action changes in historical observations. The model, conditioned on language instructions and image observations with the causal transformer structure, is supervised by the noise that we add to the continuous actions. In other words, we conduct the diffusion objective directly in the action chunk space with a large transformer model, differently from a shared diffusion action head with a few MLP layers [69].

The proposed Diffusion Transformer Policy is a general design that can be scaled to different datasets, and demonstrates excellent performance. Meanwhile, we can also add additional observation tokens and input into the transformer structure. Appendix A provides more details.

3.2. Training Objective

In our architecture, the denoising network $\epsilon_\theta(\boldsymbol{x}^t, c_{obs}, c_{instru}, t)$ is the entire causal transformer, where c_{obs} is the image observation, c_{instru} is the language instruction, and $t \in 1, 2, ...T$ is the step index in our experiments. During the training stage, we sample a Gaussian noise vector $\boldsymbol{x}^t \in \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ at timestep t, where T is the number of denoising timesteps, and add it to action $\boldsymbol{a}$ as $\hat{\boldsymbol{a}}$ to construct the noised action token, finally predicting the noise vector $\hat{\boldsymbol{x}}$ based on the denoising network $\epsilon_\theta(\hat{\boldsymbol{a}}, c_{obs}, c_{instru}, t)$, where t is randomly sampled during training. We optimize the network with MSE loss between $\boldsymbol{x}^t$ and $\hat{\boldsymbol{x}}^t$.

To generate an action, we apply T steps of denoising with the optimized transformer architecture ϵ_θ from a sampled gaussian noise vector $\boldsymbol{x}^T$ as follows,

$$\boldsymbol{x}^{t-1} = \alpha(\boldsymbol{x}^t - \gamma\epsilon_\theta(\boldsymbol{x}^t, c_{obs}, c_{instru}, t) + \mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I})). \quad (1)$$

where α, γ, σ is the noise scheduler [21]. In our experiments, ϵ_θ is to predict the noise that adds to the action.

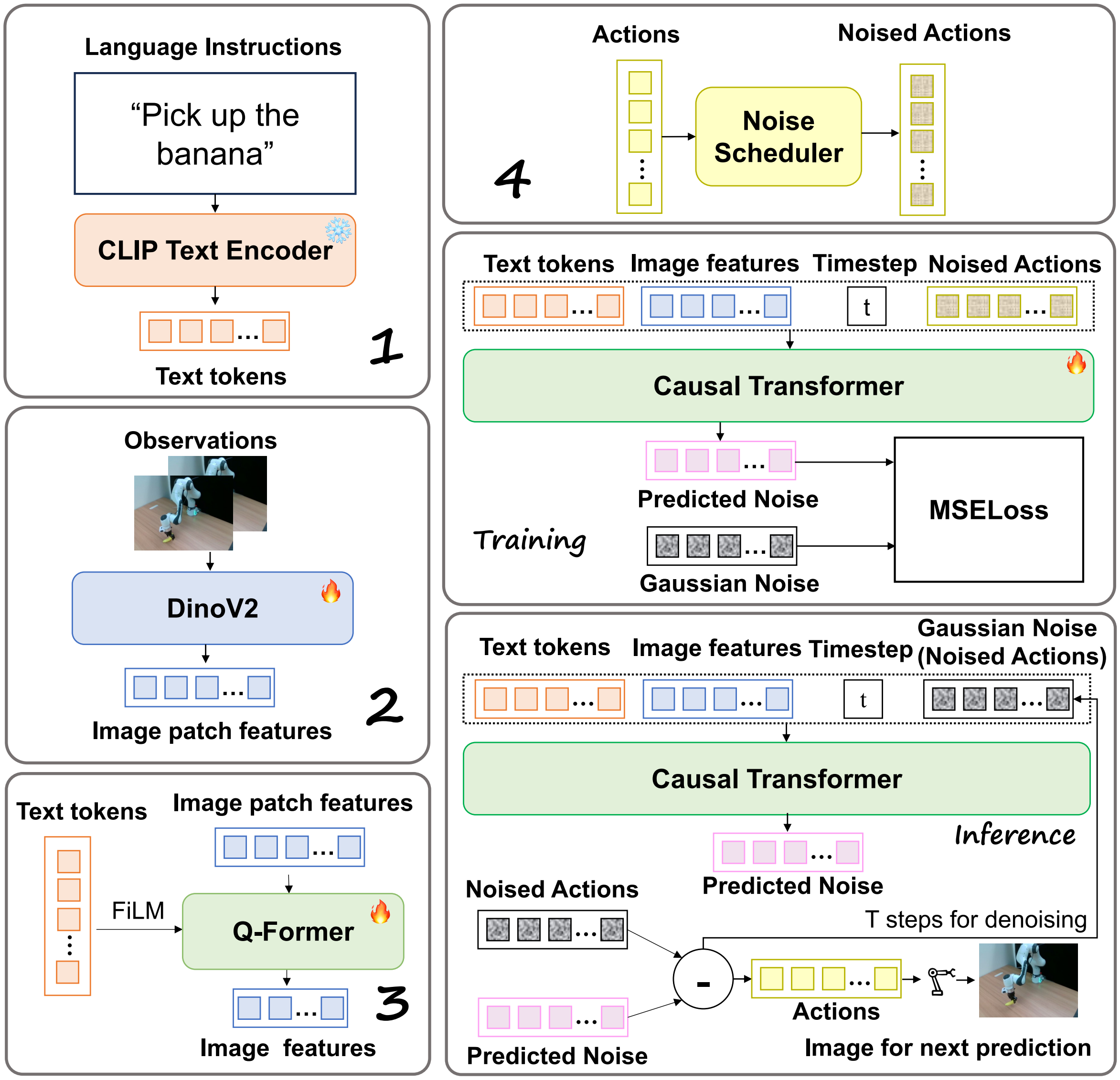


Figure 3. Our model is a Transformer diffusion structure. The model first incorporates a pretrained CLIP network to obtain instruction tokens. Meanwhile, we use the DINO-V2 [49] model to encode image observations, followed by a Q-Former to query observation features for each image observation. Next, we concatenate instruction tokens, image observation features, timestep, and noised action together to construct a token sequence as the input for transformer network to denoise the raw actions.

3.3. Pretraining Data

To evaluate the proposed Diffusion Transformer Policy, we choose Open X- Embodiment datasets [50] for pretraining the model. We mainly follow [28, 69] to choose the datasets and set the weights for each dataset. We normalize the actions similar to [50] and filter out outlier actions in the dataset. Additional details are provided in Appendix B.

3.4. Pretraining Details

We devise the proposed Diffusion Transformer architecture and evaluate the pretraining approach in the large cross-embodiment datasets [50]. We use the DDPM [21] diffusion objective in the pretraining stage with $T = 1000$ for the Open X-Embodiment dataset [50], while we set $T = 100$ with DDIM [66] for zero-shot evaluation to accelerate the inference. According to the preliminary experiment from Maniskill2 [18], we use 2 observation images and predict 32 action chunks. We train the network with AdamW [38] by 100,000 steps. We set the learning rate of the casual transformer and Q-Former as 0.0001, the learning rate of DINOv2 as 0.00001, and the batch size as 8902. More pretraining details are provided in the Appendix A.

Table 1. Comparison with RT-1-X [6], Octo-base [69] and OpenVLA-7B [28] on SimplerEnv (average variance and matching results of Google Robot [6]). The results are reported as success rate.

Method	coke_can		move_near		drawer	
	match	variant	match	variant	match	variant
RT-1-X [6]	56.7%	49.0%	31.7%	32.3%	**59.7%**	29.4%
Octo-Base [69]	17.0%	0.6%	4.2%	3.1%	22.7%	1.1%
OpenVLA-7B [28]	16.3%	54.5%	46.2%	47.7%	35.6%	17.7%
DiT Policy(Ours)	**72.7%**	**60.0%**	**56.7%**	**57.5%**	46.3%	**37.5%**

4. Experiments

We evaluate the proposed methods with two baselines in three environments. We leverage Maniskill2 to present the ability of Diffusion Transformer Policy on large scale novel view generalization. Meanwhile, we demonstrate the generalization of the pretrained Diffusion Transformer Policy on CALVIN benchmark. Lastly, we further show the generalization of DiT Policy on Real Franka Arm.

4.1. Baselines

Discretization Action Head We implement the RT-1 [6] style baseline models with a similar structure as ours. We keep the Instruction Tokenization and Image backbone. Different from ours, we discretize each dimension of the action into 256 bins [6], and leverage the transformer network to predict the action bin indexes. Following [6, 7], we use cross-entropy loss to optimize the network.

Diffusion Action Head We also implement a diffusion action head strategy [69]. Specifically, we utilize a three-layer MLP network as our denoising network condition on the output of each action token embedding by the same transformer architecture as ours. Notably, this baseline has more parameters (the additional MLP) compared to DiT policy.

4.2. SimplerEnv

SimplerEnv [34] is a Real-to-Sim platform for evaluating the policy learned from real robot data with a simulation platform. In this section, we compare with the popular generalist policy trained on Open X-embodiment dataset, including RT-1-X [6, 50], Octo [69] and OpenVLA [28], on Google Robot Simulation with different variances. We follow the evaluation from SimplerEnv [34] for a fair comparison, which includes “pick up coke can”, “move an object near to others”, “open drawer”, “close drawer”.

Table 1 demonstrates that the proposed approach achieves a strong generalization performance under different variances, including background, texture, objects, spatial positions, and so on. The variance experiments show the robust of Diffusion Transformer Policy.

4.3. CALVIN

CALVIN (Composing Actions from Language and Vision) [43] is an open-source simulated benchmark to learn long-horizon language-conditioned tasks. CALVIN [43] includes four different scenes tagged as ABCD and presents a novel scene evaluation benchmark, ABC→D, *i.e.*, trained on environments A, B, and C and evaluated on environment D. The goal of CALVIN is to solve up to 1000 unique sequence chains with 5 distinct subtasks. The benchmark requires successfully solving the task sequence with 5 continuous subtasks, and one of the important evaluation indicators is the success sequence length.

Setup. In this section, we utilize CALVIN (ABC→D) to evaluate the novel task generalization of Diffusion Transformer Policy architecture. Specifically, we directly apply the proposed method on CALVIN with a single static RGB camera and predict the end-effector action, including 3 dimensions for translation, 3 dimensions for Euler angles rotation and 1 dimension for gripper position (open or close). We evaluate Diffusion Transformer Policy and Diffusion Action Head [69] on CALVIN, and leverage the pretrained model on Open X-Embodiment to initialize the model for CALVIN.

Optimization Details. While training Calvin, 2 history images are used as input. For each iteration, the model predicts 10 future frames supervised by MSE loss. An AdamW optimizer is used together with a decayed learning rate with half-cycle cosine scheduler after several steps of warming up. The learning rate is initialized as 1e-4. We use 4 NVIDIA A100 GPUs(80GB) to train the model for 15 epochs with a global batch size of 128.

Comparisons. Table 2 presents the comparisons with previous methods on Calvin and the proposed methods. Without whistles and bells, the proposed Diffusion Transformer Policy achieves the state-of-the-art results. Particularly, we only use RGB camera stream for observation. The superior demonstrates the effectiveness of Diffusion Transformer Policy. Meanwhile, the pretraining on Open X-Embodiment Datasets significantly facilitates the performance by 1.23, which demonstrates the transferability of Diffusion Transformer Policy. By contrast, the performance of diffusion action head is worse than Diffusion Transformer Policy by 0.45, though we load similar pretraining weights for diffusion head architecture. DiT Policy can perceive visual subtle nuances for long-horizon tasks, and scale across different environments, *e.g.*, transferring the knowledge from the diverse real datasets to the CALVIN dataset.

4.4. LIBERO

LIBERO is a general benchmark focused on knowledge transfer in multitask and lifelong robot learning problems[36]. This benchmark consists of four sub-datasets, which are called LIBERO-SPATIAL, LIBERO-

Table 2. The comparisons with state-of-the-art approaches on Calvin Benchmark under success rate and average success length. MDT [57] is from issue 9 of its GitHub repo and GR-MG [31]. 'S-RGB' indicates Static RGB, 'G-RGB' indicate Gripper RGB. 'S-RGBD' indicate Static RGB-D, 'G-RGBD' indicates Gripper RGB-D, 'P' is the observation arm position 'Proprio', 'Cam' indicates camera parameters. 'Pretraining' indicates pretraining dataset or multi-modal model.

Method	Input	No. Instructions in a Row (1000 chains)					
		1	2	3	4	5	Avg.Len.
MDT* [57]		61.7%	40.6%	23.8%	14.7%	8.7%	1.54
SPIL [80]	S-RGB,G-RGB	74.2%	46.3%	27.6%	14.7%	8.0%	1.71
RoboFlamingo [33]	S-RGB,G-RGB	82.4%	61.9%	46.6%	33.1%	23.5%	2.47
SuSIE [3]	S-RGB	87.0%	69.0%	49.0%	38.0%	26.0%	2.69
GR-1 [73]	S-RGB,G-RGB,P	85.4%	71.2%	59.6%	49.7%	40.1%	3.06
3D Diffuser [26]	S-RGBD,G-RGBD,P,Cam	92.2%	78.7%	63.9%	51.2%	41.2%	3.27
diffusion head w/o pretrain	S-RGB	75.5%	44.8%	25.0%	15.0%	7.5%	1.68
diffusion head	S-RGB	94.3%	77.5%	62.0%	48.3%	34.0%	3.16
Ours w/o pretrain	S-RGB	89.5%	63.3%	39.8%	27.3%	18.5%	2.38
Ours	S-RGB	**94.5%**	**82.5%**	**72.8%**	**61.3%**	**50.0%**	**3.61**

Table 3. Comparison with Diffusion Policy[12], Octo[69] and OpenVLA[28] on LIBERO[36]. Besides the results of our model, all of the other results are from [28]. The results are a little different with we reported in the paper, this is because that we find a more suitable scheduler for finetuning and make a little increasing on the performance.

Method	**LIBERO-SPATIAL**	**LIBERO-OBJECT**	**LIBERO-GOAL**	**LIBERO-LONG**	**Average**
Diffusion Policy from scratch[12]	78.3%	92.5%	68.3%	50.5%	72.4%
Octo fine-tuned[69]	78.9%	85.7%	84.6%	51.1%	75.1%
OpenVLA fine-tuned[28]	**84.9%**	88.4%	79.2%	53.7%	76.5%
DiT Policy fine-tuned(Ours)	84.2%	**96.3%**	**85.4%**	**63.8%**	**82.4%**

OBJECT, LIBERO-GOAL and LIBERO-100, respectively. These sub-datasets are used to test different abilities of the models. LIBERO-SPATIAL tests the ability of spatial relationship understanding, containing data of the same object set but different layouts. LIBERO-OBJECT tests the ability of object transferring, containing data of the same layouts but different object sets. LIBERO-GOAL tests the ability of task understanding and transferring, containing data of the same object sets and layouts but different tasks. LIBERO-100 can further divide into LIBERO-90 and LIBERO-10(Also called LIBERO-LONG), which are used for policy pretraining and long-horizon tasks testing, respectively. LIBERO-100 consists of data of diverse objects, layouts and backgrounds. In the latest version of OpenVLA[28], they compares the performance of several models on LIBERO. We follow OpenVLA and make a comparison with these models. We first pretrain our model on the Open X-Embodiment[50] then full-finetune it on LIBERO. We use the dataset that slightly modified by OpenVLA[28] in order to show a fair comparison. The results are shown in Table 3. Besides the results of our model, all of the other results are from [28]. We achieve a better performance on most of the sub-datasets of LIBERO and make a nearly 6% increasing on the average success rate. Specially, we make great progress on the LIBERO-LONG, which means a great potential for completing the long-horizon tasks of our model.

4.5. Maniskill2

Maniskill2 [18] is the next generation of the SAPIEN Maniskill benchmark [44], which is widely used to evaluate the generalized manipulation ability of the embodied models. It contains 20 different manipulation tasks families and over 4 million demonstration frames with different settings, including rigid/soft body, single/dual arm, etc. Maniskill2 also provides a fast and easy way to change the camera view and replay the trajectories. It is useful for the researchers working on generalized policy.

Setup. In our experiments, we select 5 tasks (PickCube-v0, StackCube-v0, PickSingleYCB-v0, PickClutterYCB-v0, PickSingleEGAD-v0) from Maniskill2, and then construct a camera pool with 300,000 random cameras, then sample 20 cameras from the camera pool to render a trajectory each time, and finally obtain about 40K trajectories totally. Given that the number of trajectories is large, we thus train the network on Maniskill2 from scratch. Moreover, we split the dataset into training set and validation set according to a ratio of 19:1. During the splitting, it is guaranteed that the single trajectory rendered under different camera views will appear in either training set or validation set in order to avoid data leaking. Specially, there are 74 different categories to pick and place in the task family PickSingleYCB-v0. In addition to the mentioned rules, we ensure each category can be found in both training set and validation set. After that, we sample 100 trajectories for each task family randomly from the validation set, constructing a close loop evaluation dataset with 500 trajectories in total. While training, considering the balance between different task families, we adjust the number of data pieces to the same by simply copying the trajectories from task family with fewer trajectories originally. The ability of the model is measured by the success rate of each task family while executing the close loop evaluation dataset.

Optimization Details. We optimize the network with AdamW [38] by 50,000 steps on Maniskill2 and we set the learning rate as 0.0001. The number of training timesteps T is 100 in Maniskill2 and the global batch size is 1024.

Comparisons. Table 10 compares the proposed method with discretized action head [6, 7] and diffusion action head [69]. The experiments demonstrate Diffusion Transformer Policy achieves better results compared to Discretization Action Head strategy [6] under the large scale novel view maniskill2 benchmark. Meanwhile, Diffusion Transformer Policy demonstrates better performance in more complex tasks, *e.g*., Diffusion Transformer Policy improves diffusion action head [69] by 20% in task PickSingleYCB and by 12% task PickClutterYCB. Those experiments show that Diffusion Transformer Policy achieves better scalability in the large scale diverse datasets, and meanwhile achieves better generation in camera view generalization.

4.6. Real Franka Arm

We finally evaluate the proposed method by pretraining on Open X-Embodiment Datasets [50], and evaluate it in our Franka Arm environment under zero-shot generalization, 10-shot generalization, and in domain finetuning generalization. We meanwhile compare it by finetuning on Libero [36].

Setup. We set up the franka on the table with a black background. Meanwhile, we use a single third-person RGB camera about 1.5 meters away from the Franka Arm. Please refer to Figure 4 for the visualized demonstration. Considering that the environment of our Franka setup is different from the scenes in the pretraining data Open X-Embodiment [68], we mainly evaluate the proposed method on out-of-the-box generation and few-shot generation. In our experiments, we evaluate each model with the same scene, and the object is placed in 9 similar positions in a 9-grid format in front of the franka arm. Meanwhile, we maintain a small variance in those positions placing the objects for evaluation.

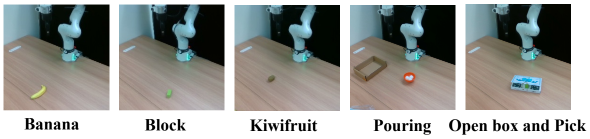


Figure 4. Illustration of Franka environment and exampled tasks.

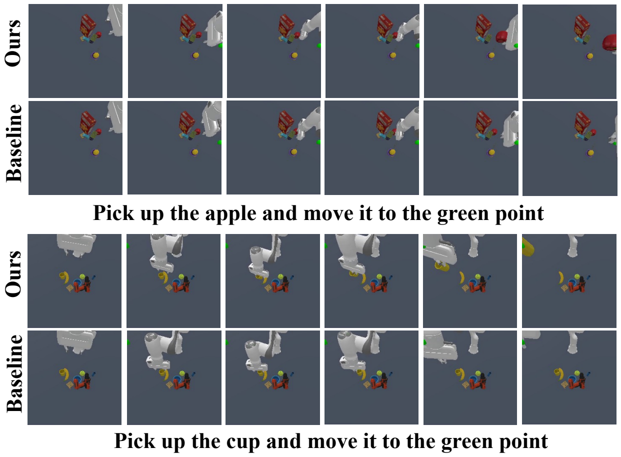
To evaluate the ability of quick finetuning with a few training samples, we set two pick and place tasks, and further collect five complex manipulation tasks, *i.e*., ‘Pick up the bowl and pour the balls into the box’, ‘Open the box and pick up the block within the box’, ‘pick up the banana and move it into the pen container’, ‘stack the bowls’, and ‘pick up the cup and pouring the coffee beans from the cup into the bowl’. We collect 10 samples for each task for *10-shot finetuning generalization*. Figure 4 presents the scenes and tasks. The image in Figure 4 is the model input. Our real-world environment is challenging since the object (less than 20mm) is small compared to the whole scene. We also present a simple evaluation for in-domain evaluation for comparing different action strategies under in-domain generalization with 5 pick and place tasks, including ‘green block’, ‘kiwifruit’, ‘banana’, ‘the tiny green block’, ‘pink block’. Meanwhile, we collect 50 trajectories for each task in the first three tasks, while leaving the remaining two tasks (‘Pick up the tiny green block’ and ‘Pick up the pink block’) for out-of-distribution evaluation.

Finetuning details. In our experiments, we finetune the proposed method on the real Franka Arm with Lora [22] and AdamW [38] for 10,000 steps. We set the number of timesteps as 100 for DDPM [21], and batch size as 512.

10-shot Finetuning Generalization We directly take the models pretrained on Open X-Embodiment to evaluate 10-shot generalization in our environments. Here, to maintain the consistency with OpenVLA [28], we finetune the network with one observation and one step prediction. We compare the proposed method with OpenVLA [28] and Octo-base [69] models. Besides, we evaluate it on long horizon tasks. Table 5 presents the proposed Dit Policy archives consistent improvement compared to Octo and OpenVLA. Specifically, the proposed DiT policy demonstrates slight improvement on Pickup tasks, while the proposed DiT policy can achieve clear better performance on all more complex tasks. For those long-horizon tasks, we can find OpenVLA can complete the first task effectively while failing to under- stand the long-horizon task. For example, it completely misunderstands the insert operation, which we illustrate in the demo video. Differently, Octo is better at completing tasks with rotations and approaching the second tasks as il- lustrated with the Demo video. By contrast, the proposed DiT policy is more robust, and achieves clearly better perfor- mance in long-horizon tasks.

Table 4. Comparision with two baseline methods on Maniskill2 under success rate. SingleYCB indicates PickSingleYCB, ClutterYCB indicates PickClutterYCB, SingleEGAD indicates PickSingleEGAD. Disc ActionHead indicates Discretized Action Head strategy [6], while Diff ActionHead shows Diffusion Action Head [69].

Method	**All**	**PickCube**	**StackCube**	**SingleYCB**	**ClutterYCB**	**SingleEGAD**
Disc ActionHead	30.2%	41.0%	33.0%	22.0%	1.0%	54.0%
Diff ActionHead	58.6%	**86.0%**	76.0%	37.0%	24.0%	70.0%
DiT Policy(ours)	**65.8%**	79.0%	**80.0%**	**62.0%**	**36.0%**	**72.0%**



Pick up the apple and move it to the green point

Pick up the cup and move it to the green point

Figure 5. Visualized comparison between Diffusion Transformer Policy and Diffusion Action Head baseline on Maniskill2 (PickClutterYCB). The first raw is Diffusion Tranformer Policy, while the second raw is the baseline method with Diffusion Action Head.

We provide a visualized analysis in Appendix C.1.

In domain Finetuning Generalization Table 6 presents the performance of the proposed DiT Policy compared to baseline methods. We observe different objects demonstrate various performances according to their attributes. The banana is the easiest object to pick up because the banana is longer, while kiwifruit is fat compared to other objects and all models achieve poor performance. The proposed DiT Policy effectively improves the diffusion action head according to Table 6. We find the discretized action head baseline achieves poor performance. Meanwhile, the DiT Policy is still able to pick up the novel object (*e.g.*, the pink object) with a low success rate, while the baseline methods totally fail.

4.7. Ablation Study

In this section, we ablate some of the important designs of the model architecture, including the length of horizon, the length of observation, execution steps for evaluation on Maniskill2.

Trajectory length. The length of action chunks has an important effect on the performance of different tasks. Table 7 shows that performance increases with increasing trajectory length. Meanwhile, we notice the performance of more complex tasks, *e.g.*, PickClutterYCB, increases significantly with increasing trajectory length, while the easy task, *e.g.*, PickCube, maintains high performance after the trajectory length is greater than 4. Meanwhile, the long horizon optimization significantly facilitates the performance since long horizon optimization is able to provide the target object position and help the model understand the localization of the object. For example, task PickClutterYCB with multiple YCB objects, requires the model to understand which one is the corresponding object.

Observation length. In our experiments, we find the length of history observation images also significantly affects the performance. At first, the performance significantly drops when we increase the length of observation history to 3. It might because it is more difficult for the model to converge with more observations since the number of corresponding image tokens also increases. Secondly, we observe using two image observations is more helpful for the performance when the prediction horizon is long. For example, when the length of trajectory is 32, the experiment with two observations achieves better performance. We think two observations can provide the visualized difference between two positions, and the difference of continunous gripper position indicates the action. The visualized difference is beneficial for future action prediction. However, for short horizon, the model majorly learns the projection from current observation to the corresponding actions.

Execution steps. Since the proposed model is able to predict multiple future actions, we can execute multiple steps in one inference. Here, we ablate the effect of execution steps under a model with trajectory length 32 in Table 8. The ablation study shows that the short execution steps are slightly better longer execution steps, *i.e.*, the farther away from the current frame, the worse the prediction quality.

4.8. Visualized Comparison

Maniskill2. The proposed Diffusion Transformer Policy is able to model better action sequences. We conduct visual-

Table 5. Comparision with Octo and Openvla on Real Franka Arm under 10-shot finetuning. Results are reported as success rate. 10-shot finetuning indicates we finetune the model with only 10 samples for each task. ‘Pourballs’ indicates pick up the bowl and pour the balls into the box. ‘Open & Pick’ indicates ‘open the box (Step-1) and pick up the block(Step-2) from the box’. ‘Pick&Insert’ indicates ‘pick up the banana (Step-1) and insert into the small pen container (Step-2)’. ‘Stacking Bowls’ is ‘stack the three bowls’, in which ‘Step-1’ is the first tacking and ‘Step-2’ is the second tacking. ‘Pick& Pouring’ is picking the mark cup (Step-1) and pouring the coffee beans from the cup into the bowl (Step-2).

Method	PickBlock	PickBanana	Pourballs	Open & Pick		Stacking Bowls		Pick & Insert		Pick & Pouring	
				Step-1	Step-2	Step-1	Step-2	Step-1	Step-2	Step-1	Step-2
Octo [69]	7.4%	22.2%	14.8%	22.2%	0	10%	0	20%	0	10%	0
OpenVLA [28]	7.4%	33.3%	18.5%	7.4%	0	40%	0	80%	0	20%	0
Ours	**14.8%**	**33.3%**	**29.6%**	**29.6%**	**7.4%**	**50%**	**12%**	**90%**	**10%**	40%	10%

Table 6. Comparision on Real-Franka tasks with few-shot finetuning about different action strategies. Discretized indicates discretized action head and Diffusion indicates Diffusion action head. We represent the values in success rate. Block is ‘Pick up the green block’, Banana is ‘Pick up the banana’, Kiwi is ‘Pick up the Kiwifruit’, Tiny is unseen object task ‘Pick up the green tiny block’, Oval is unseen object task ‘Pick up the pink oval block’.

Method	All	Block	Banana	Kiwi	Tiny	Oval
Discretized [6]	19.3%	29.6%	51.9%	14.8%	0	0
Diffusion [69]	34.8 %	40.7%	85.2%	25.9%	22.2%	0
DiT Policy (ours)	**46.9%**	**55.6%**	**90.3%**	**44.4%**	**37.0%**	**7.4%**

Table 7. Ablation on Maniskill2 about the number of history observation images and the length of the trajectory. #obs indicates the number of history observation images. #traj shows the length of trajectory, *i.e*., the sum length of observation and action prediction chunks. PickC indicates PickCube, StackC indicates StackCube, SingleYCB indicates PickSingleYCB, ClutterYCB indicates PickClutterYCB, EGAD indicates PickSingleEGAD.

#obs	#traj	All	PickC	StackC	SingleYCB	ClutterYCB	EGAD
2	2	40.8%	68.0%	54.0%	33.0%	9.0%	40.0%
2	4	51.6%	81.0%	69.0%	44.0%	11.0%	53.0%
2	8	62.4%	**89.0** %	78.0%	54.0%	25.0%	66.0%
2	16	65.6%	83.0%	**80.0** %	**70.0** %	25.0%	70.0%
2	32	**65.8** %	79.0%	**80.0** %	62.0%	**36.0** %	**72.0%**
1	32	61.6%	78.0%	76.0%	64.0%	24.0%	66.0%
1	1	51.0%	79.0%	66.0%	42.0%	19.0%	49.0%
3	3	35.4%	54.0%	49.0%	27.0%	5.0%	42.0%

Table 8. The effect of the number of execution steps on Maniskill2. #steps indicates the number of steps that we execute each prediction.

#steps	1	2	4	8	16
All	**61.6%**	60.8 %	60.6 %	60.0 %	58.0 %

ized analysis between the proposed method and the diffusion action head baseline on Maniskill2 in Figure 9. We select two trajectories from PickClutterYCB task, which is the most challenging task in Maniskill2. Figure 9 presents the grasp position is significantly important for picking up successfully, and the main reason that the baseline fails to pick up is the wrong grasp position. Meanwhile, we observe the major challenge of task PickClutterYCB is the grasping position prediction, especially when the target object is near by other objects. Compared to the diffusion action head baseline, Diffusion Transformer Policy is able to predict better action chunks for correctly picking the object with a suitable end-effector pose.

Real Franka Arm. We also illustrate the comparison between the Diffusion Transformer Policy and diffusion action head baseline on real Franka Arm in Figure 10. We demonstrate the experimental results under few-shot finetuning setting. We find the proposed method achieves better action prediction when the Gripper is approaching the object and finally picks up the small green object successfully, while the baseline fails to pick up due to the inaccurate grasp position. Meanwhile, we observe that failures are usually caused by a tiny position bias and we can not even directly discriminate the position by eyes from the image. For those cases, we argue the diffusion transformer policy has learnt better grasp position during the pretraining stage, and thus reduce failure rate due to the wrong grasp pose, while it is difficult for the diffusion action head. We demonstrates more comparisons to analyze the challenges in the Franka Arm.

5. Conclusion and Discussion

In this paper, we present a Diffusion Transformer architecture for generalist robot learning, named as Diffusion Transformer Policy. Diffusion Transformer Policy directly utilizes the large transformers as a denoising network to de-

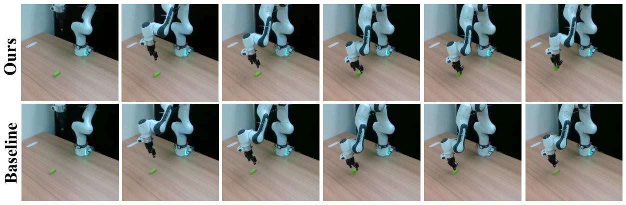


Figure 6. Visualized comparison between Diffusion Transformer Policy and Diffusion Action Head strategy on Real Franka Arm (Pick up the green block). The first raw is DiT Policy, while the second raw is the baseline method with Diffusion Action Head.

noise the continuous actions conditioned on language instruction and image observations. The proposed architecture retains the scale attribute of the transformer, thus is capable of generalizing to different datasets with a unified architecture. Extensive experiments on SimplerEnv, Libero, Maniskill2, CALVIN, real Frank Arm demonstrate the effectiveness of the proposed method. Particularly, the proposed approach achieves state-of-the-art performance in CALVIN (ABC→D) with only a single observation, and achieves consistent improvement compared to OpenVLA and Octo on SimplerEnv.

A limitation of the Diffusion Transformer Policy is that it requires multiple denoising steps during inference, which will impede the inference speed in the real application. In this paper, we focus on the modeling of complex and diverse actions. We think it is possible to improve the finetuning strategy with a few denoising steps to accelerate the inference speed.

References

[1] Suneel Belkhale, Yuchen Cui, and Dorsa Sadigh. Hydra: Hybrid robot actions for imitation learning. In *Conference on Robot Learning*, pages 2113–2133. PMLR, 2023. 3

[2] Homanga Bharadhwaj, Jay Vakil, Mohit Sharma, Abhinav Gupta, Shubham Tulsiani, and Vikash Kumar. Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4788–4795. IEEE, 2024. 2

[3] 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. 6

[4] Denis Blessing, Onur Celik, Xiaogang Jia, Moritz Reuss, Maximilian Li, Rudolf Lioutikov, and Gerhard Neumann. Information maximizing curriculum: A curriculum-based approach for learning versatile skills. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[5] Konstantinos Bousmalis, Giulia Vezzani, Dushyant Rao, Coline Devin, Alex X Lee, Maria Bauza, Todor Davchev, Yuxiang Zhou, Agrim Gupta, Akhil Raju, et al. Robocat: A self-improving foundation agent for robotic manipulation. *arXiv preprint arXiv:2306.11706*, 2023. 2

[6] 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. 1, 2, 3, 5, 7, 8, 9

[7] 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. 1, 2, 3, 5, 7

[8] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024. 2

[9] Jiahang Cao, Qiang Zhang, Jingkai Sun, Jiaxu Wang, Hao Cheng, Yulin Li, Jun Ma, Yecheng Shao, Wen Zhao, Gang Han, et al. Mamba policy: Towards efficient 3d diffusion policy with hybrid selective state models. *arXiv preprint arXiv:2409.07163*, 2024. 2

[10] Boyuan Chen, Diego Marti Monso, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *arXiv preprint arXiv:2407.01392*, 2024. 2

[11] Lili Chen, Shikhar Bahl, and Deepak Pathak. Playfusion: Skill acquisition via diffusion from language-annotated play. In *Conference on Robot Learning*, pages 2012–2029. PMLR, 2023. 2

[12] 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. 2, 6, 3

[13] Zichen Jeff Cui, Yibin Wang, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. From play to policy: Conditional behavior generation from uncurated robot data. *arXiv preprint arXiv:2210.10047*, 2022. 3

[14] Shivin Dass, Jullian Yapeter, Jesse Zhang, Jiahui Zhang, Karl Pertsch, Stefanos Nikolaidis, and Joseph J. Lim. Clvr jaco play dataset, 2023. 3

[15] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 2

[16] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023. 2

[17] Hao-Shu Fang, Hongjie Fang, Zhenyu Tang, Jirong Liu, Junbo Wang, Haoyi Zhu, and Cewu Lu. Rh20t: A robotic dataset for learning diverse skills in one-shot. In *RSS 2023 Workshop on Learning for Task and Motion Planning*, 2023. 2

[18] 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. 4, 6

[19] Huy Ha, Pete Florence, and Shuran Song. Scaling up and distilling down: Language-guided robot skill acquisition. In *Conference on Robot Learning*, pages 3766–3777. PMLR, 2023. 2

[20] Minho Heo, Youngwoon Lee, Doohyun Lee, and Joseph J Lim. Furniturebench: Reproducible real-world benchmark for long-horizon complex manipulation. *arXiv preprint arXiv:2305.12821*, 2023. 3

[21] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2, 3, 4, 7

[22] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 7

[23] Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puhao Li, Yan Wang, Qing Li, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3d world. *arXiv preprint arXiv:2311.12871*, 2023. 2

[24] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, and Sergey Levine. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation, 2018. 3

[25] Siddharth Karamcheti, Suraj Nair, Annie S Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-driven representation learning for robotics. *arXiv preprint arXiv:2302.12766*, 2023. 2

[26] Tsung-Wei Ke, Nikolaos Gkanatsios, and Katerina Fragkiadaki. 3d diffuser actor: Policy diffusion with 3d scene representations. *arXiv preprint arXiv:2402.10885*, 2024. 2, 6

[27] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024. 3

[28] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024. 1, 2, 3, 4, 5, 6, 7, 9

[29] Vikash Kumar, Rutav Shah, Gaoyue Zhou, Vincent Moens, Vittorio Caggiano, Abhishek Gupta, and Aravind Rajeswaran. Robohive: A unified framework for robot learning. *Advances in Neural Information Processing Systems*, 36, 2024. 3

[30] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023. 3

[31] Peiyan Li, Hongtao Wu, Yan Huang, Chilam Cheang, Liang Wang, and Tao Kong. Gr-mg: Leveraging partially-annotated data via multi-modal goal-conditioned policy. *IEEE Robotics and Automation Letters*, 2025. 6

[32] Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *arXiv preprint arXiv:2406.11838*, 2024. 2

[33] Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, et al. Vision-language foundation models as effective robot imitators. *arXiv preprint arXiv:2311.01378*, 2023. 6

[34] Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. Evaluating real-world robot manipulation policies in simulation. *arXiv preprint arXiv:2405.05941*, 2024. 1, 5

[35] Zhixuan Liang, Yao Mu, Hengbo Ma, Masayoshi Tomizuka, Mingyu Ding, and Ping Luo. Skilldiffuser: Interpretable hierarchical planning via skill abstractions in diffusion-based task execution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16467–16476, 2024. 2

[36] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 5, 6, 7

[37] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023. 1

[38] I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 4, 7

[39] Corey Lynch and Pierre Sermanet. Language conditioned imitation learning over unstructured data. *arXiv preprint arXiv:2005.07648*, 2020. 2

[40] Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Conference on robot learning*, pages 1113–1132. PMLR, 2020. 2

[41] Corey Lynch, Ayzaan Wahid, Jonathan Tompson, Tianli Ding, James Betker, Robert Baruch, Travis Armstrong, and Pete Florence. Interactive language: Talking to robots in real time. *IEEE Robotics and Automation Letters*, 2023. 3

[42] Oier Mees, Lukas Hermann, and Wolfram Burgard. What matters in language conditioned robotic imitation learning over unstructured data. *IEEE Robotics and Automation Letters*, 7(4):11205–11212, 2022. 2

[43] Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3): 7327–7334, 2022. 1, 5

[44] Tongzhou Mu, Zhan Ling, Fanbo Xiang, Derek Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations. *arXiv preprint arXiv:2107.14483*, 2021. 6

[45] Vivek Myers, Andre Wang He, Kuan Fang, Homer Rich Walke, Philippe Hansen-Estruch, Ching-An Cheng, Mihai Jalobeanu, Andrey Kolobov, Anca Dragan, and Sergey Levine. Goal representations for instruction following: A semi-supervised language interface to control. In *Conference on Robot Learning*, pages 3894–3908. PMLR, 2023. 2

[46] OpenAI. Dall·e: Creating images from text, 2021. 1

[47] OpenAI. Chatgpt, 2022.

[48] OpenAI. Gpt-4: Multimodal capabilities, 2023. 1

[49] 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. 3, 4

[50] Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023. 2, 4, 5, 6, 7

[51] Tim Pearce, Tabish Rashid, Anssi Kanervisto, Dave Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, et al. Imitating human behaviour with diffusion models. *arXiv preprint arXiv:2301.10677*, 2023. 2

[52] 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. 2, 3

[53] 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*, 2018. 3

[54] Gabriel Quere, Annette Hagengruber, Maged Iskandar, Samuel Bustamante, Daniel Leidner, Freek Stulp, and Jörn Vogel. Shared control templates for assistive robotics. In *2020 IEEE international conference on robotics and automation (ICRA)*, pages 1956–1962. IEEE, 2020. 3

[55] 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. 3

[56] Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. Goal-conditioned imitation learning using score-based diffusion policies. *arXiv preprint arXiv:2304.02532*, 2023. 2

[57] Moritz Reuss, Ömer Erdinç Yağmurlu, Fabian Wenzel, and Rudolf Lioutikov. Multimodal diffusion transformer: Learning versatile behavior from multimodal goals. In *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA 2024*, 2024. 2, 6

[58] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Tim Schmidt. Stable diffusion: High-resolution image synthesis with latent diffusion models, 2021. 1

[59] 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. 2

[60] Paul Maria Scheikl, Nicolas Schreiber, Christoph Haas, Niklas Freymuth, Gerhard Neumann, Rudolf Lioutikov, and Franziska Mathis-Ullrich. Movement primitive diffusion: Learning gentle robotic manipulation of deformable objects. *IEEE Robotics and Automation Letters*, 2024. 2

[61] Nur Muhammad Mahi Shafiullah, Anant Rai, Haritheja Etukuru, Yiqian Liu, Ishan Misra, Soumith Chintala, and Lerrel Pinto. On bringing robots home. *arXiv preprint arXiv:2311.16098*, 2023. 3

[62] Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. Gnm: A general navigation model to drive any robot. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7226–7233. IEEE, 2023. 1, 2

[63] Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. Vint: A foundation model for visual navigation. *arXiv preprint arXiv:2306.14846*, 2023. 1, 2

[64] Rutav Shah, Roberto Martín-Martín, and Yuke Zhu. Mutex: Learning unified policies from multimodal task specifications. *arXiv preprint arXiv:2309.14320*, 2023. 2

[65] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023. 2

[66] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 4

[67] Ajay Sridhar, Dhruv Shah, Catherine Glossop, and Sergey Levine. Nomad: Goal masked diffusion policies for navigation and exploration. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 63–70. IEEE, 2024. 2

[68] Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Sean Kirmani, Brianna Zitkovich, Fei Xia, et al. Open-world object manipulation using pre-trained vision-language models. *arXiv preprint arXiv:2303.00905*, 2023. 7

[69] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024. 1, 2, 3, 4, 5, 6, 7, 8, 9

[70] 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. 3

[71] Yixiao Wang, Yifei Zhang, Mingxiao Huo, Ran Tian, Xiang Zhang, Yichen Xie, Chenfeng Xu, Pengliang Ji, Wei Zhan, Mingyu Ding, et al. Sparse diffusion policy: A sparse, reusable, and flexible policy for robot learning. *arXiv preprint arXiv:2407.01531*, 2024. 2

[72] 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. 2

[73] 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. 6

[74] Zhou Xian, Nikolaos Gkanatsios, Theophile Gervet, and Katerina Fragkiadaki. Unifying diffusion models with action detection transformers for multi-task robotic manipulation. In *7th Annual Conference on Robot Learning*, page 5, 2023. 2

[75] Ted Xiao, Harris Chan, Pierre Sermanet, Ayzaan Wahid, Anthony Brohan, Karol Hausman, Sergey Levine, and Jonathan Tompson. Robotic skill acquisition via instruction augmentation with vision-language models. *arXiv preprint arXiv:2211.11736*, 2022. 2

[76] Jonathan Yang, Catherine Glossop, Arjun Bhorkar, Dhruv Shah, Quan Vuong, Chelsea Finn, Dorsa Sadigh, and Sergey Levine. Pushing the limits of cross-embodiment learning for manipulation and navigation. *arXiv preprint arXiv:2402.19432*, 2024. 2

[77] 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 *ICRA 2024 Workshop on 3D Visual Representations for Robot Manipulation*, 2024. 2

[78] Edwin Zhang, Yujie Lu, William Wang, and Amy Zhang. Language control diffusion: Efficiently scaling through space, time, and tasks. *arXiv preprint arXiv:2210.15629*, 2022. 2

[79] Gaoyue Zhou, Victoria Dean, Mohan Kumar Srirama, Aravind Rajeswaran, Jyothish Pari, Kyle Hatch, Aryan Jain, Tianhe Yu, Pieter Abbeel, Lerrel Pinto, et al. Train offline, test online: A real robot learning benchmark. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9197–9203. IEEE, 2023. 3

[80] Hongkuan Zhou, Zhenshan Bing, Xiangtong Yao, Xiaojie Su, Chenguang Yang, Kai Huang, and Alois Knoll. Language-conditioned imitation learning with base skill priors under unstructured data. *IEEE Robotics and Automation Letters*, 2024. 6

[81] Xinghao Zhu, Ran Tian, Chenfeng Xu, Mingxiao Huo, Wei Zhan, Masayoshi Tomizuka, and Mingyu Ding. Fanuc manipulation: A dataset for learning-based manipulation with fanuc mate 200id robot, 2023. 3

[82] Yifeng Zhu, Abhishek Joshi, Peter Stone, and Yuke Zhu. Viola: Imitation learning for vision-based manipulation with object proposal priors. In *Conference on Robot Learning*, pages 1199–1210. PMLR, 2023. 3

Diffusion Transformer Policy

Supplementary Material

A. Additional Training and Model Details

The structure of our Model is shown in Figure 2. In this section, we will talk about some extra details of our model. The language instruction is encoded by a pretrained clip model and freeze the encoder in the training loop. We then resize the input images into 224$*$224 and feed it into a pretrained ViT model. The selected ViT is the base version of DinoV2. All the parameters in DinoV2 are trained. After the above process, we use a Q-Former to decrease the size of image features. The Q-Former is from scratched with a depth of 4. In each block, we insert the text token as a FiLM Condition to get the image features containing language information. The query length of the image features will reduce to 32 when out of the Q-Former. Then, we concatenate the processed text features and image features, together with the action nosied by a DDPM scheduler with 100 time-step. The multimodal inputs then pass through a causal Transformer and predict the added noise, which will execute on the robot arm after a series of post-processing. The Transformer is a from scratched Llama2-type model with 12 self-attention blocks. The hidden size is set to 768. All the modules mentioned are trained except the text encoder of clip. In summary, we have 334M parameters in total and 221M trained. This is pioneering to get this performance with such a small-sized model.

B. Objects and Tasks Description

The object in our experiments is usually small. The banana is 3cm width and 16cm long. The green block is 2cm width and 4cm long. The kiwifruit is very light with 3cm width and about 4.5cm long. The tiny block is only 0.5cm width and 4cm long. *p*lease notice the images (224*224) in Figure **??**, Figure **??**, Figure **??** are the input for our model. We demonstrate the pick and place task in Figure **??**. In particular, we use the teaching strategy for pick and place tasks to collect data in our experiments. Therefore, we can collect data without remote control or mouse. We further use the remote control to collect more complex data. We demonstrate the complex task and long-horizon tasks as follows,

Pick up the bowl and pour the balls into the box. This task requires to pick up the bowl and pour the balls in the bowl into the box. The bowl is much larger than the block in our experiments. Therefore, we find the model is able to achieve better performance for picking up the bowl. Meanwhile, compared to pick and place tasks, the box of this task is fixed. Thus, it is slightly easier.

Open the box and pick up the green block. In this task, we place the box below the gripper with some variations. The variation of this task is less than the pick up tasks.

Stack the bowls. In this task, we stack the two bowls in the side into the center. For this task, we evaluate it with 20 trials with bowl position variation.

Pick up the banana and move it into the pen container. This task aims to pick up the banana and insert into the small pen container. The banana is aroud 3cm width and 16cm long, while the pen container is 65cm wide. Therefore, it is very difficult to put the banana into the pen container. For this task, we evaluate it with 20 trials with different banana places.

Pick up the banana and move it into the pen container. This task aims to pick up the banana and insert into the small pen container. The banana is aroud 3cm width and 16cm long, while the pen container is 65cm wide. Therefore, it is very difficult to put the banana into the pen container. For this task, we evaluate it with 20 trials with different banana places.

Pick up the mark cup and pouring coffee beans from cup into the bowl. This task aims to pick up the mark cup, and then pouring the coffee beans from the cup into the bowl on top of the box. The mark cup is 200g. Therefore, it is very difficult to pick up it successsfully.

C. Analysis and Discussion

C.1. Comparisons and Analysis with SoTA

In our experiments, we find OpenVLA [28] achieves good pick-up performance compared Octo [69]. However, when the task requires learning rotation operation, Octo is able to achieve better performance, e.g., Open the box. We think it is because Octo predicts continuous actions compared to OpenVLA, and the diffusion policy is not sensitive to action normalizations. Differently, OpenVLA requires to discretize the actions dependent on the action statistics. In our experiments, we calculate the statistics from the 10-shot training samples (all tasks) for OpenVLA, Octo, and DiT policy. It is difficult to obtain suitable statistics for Discretization values. However, diffusion policy is able to predict the continuous actions.

C.2. Zero-shot generalization

Figure 8 presents the proposed method is able to grasp the object, while OpenVLA and Octo fails. We observe OpenVLA and Octo fail to rightly approach the right grasp position. This experiment demonstrates the proposed Diffusion Transformer structure achieves more robust policy learning compared to discretized actions or diffusion action head.

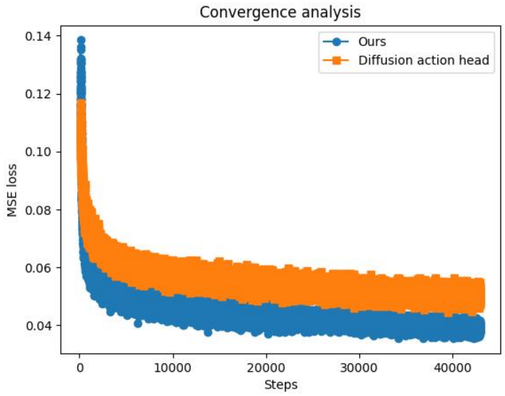


Figure 7. Convergence Analysis on OXE dataset [7]. The blue line is DiT Policy, and the orange line is Diffusion action head strategy with the same number of parameters.

Table 9. The Ablation of learning rate scheduler on Calvin Benchmark.

strategy	No. Instructions in a Row (1000 chains)					
w lr decay	**94.5%**	**82.5%**	**72.8%**	**61.3%**	**50.0%**	**3.61**
w/o lr decay	91.8%	80.0%	68.0%	56.9%	45.9%	3.43

The denoising transformer model has built a better mapping from the image observation to corresponding action chunks.

C.3. Convergence Analysis

Figure 7 presents the convergence between the baseline method and DiT Policy. We can find that the DiT policy achieves clear faster convergence compared to the diffusion action head strategy. We think this also demonstrates the scalibity of DiT Policy.

C.4. Analysis in Calvin

As illustrated in the main paper, we use a common learning rate scheduler to decay the learning rate in the experiments in Calvin, rather than a fixed learning rate of 0.0001 in our pre-training stage. We demonstrate that this can slightly improve the performance in Table 9.

D. Comparisons on More Diffusion Strategies

In our experiments, we implement a strong baseline based on the core idea of diffusion action head. However, it is slightly different from Octo [69] when we predict action chunks. Octo [69] flattens the action chunks into a single vector with a single embedding. For example, if it requires to predict 8 actions, they predict a $8 * 7 = 56$ vector. Unlike the Octo-style diffusion action head, we implement a shared diffusion action head as illustrated in Figure 1 in the main paper, which is more effective. Specially, it is more similar to the Diffussion loss [32] and Diffusion Force [10]. We further devise different diffusion heads, including Unet1D, Transformer on Calvin without pretraining.

Table 10 presents the comparisons about different baseline methods. We can find Diffusion Head with action chunks [69] achieves poor performance on Maniskill2. This indicates that the proposed diffusion action head baseline is more effective compared to the action chunks design [69]. Meanwhile, when we increase the blocks of the action head in our baseline, we do not find clean improvement compared to the three layer MLP. Moreoever, the proposed DiT policy achieves clean better performance on Maniskill2, especially on complex YCB tasks.

Table 11 shows that the proposed Dit Policy achives the better generalization on Calvin (ABC→D), compared to other diffusion head strategies [2, 12**?**].

E. Additional Details about Pretraining Data

We choose 15 large datasets from Open X-Embodiment [50] as illustrated in Table 12. We mainly follow [28, 69] to set the weights. Following [28], we further resize the image to the size of 224.

Table 10. Comparision with more diffusion baseline methods on Maniskill2 under success rate. SingleYCB indicates PickSingleYCB, ClutterYCB indicates PickClutterYCB, SingleEGAD indicates PickSingleEGAD. Disc ActionHead indicates Discretized Action Head strategy [6], while Diff ActionHead w/ action chunks is totally similar to Octo strategy [69]. Diffusion Action Head is our baseline. Diffusion Action Head w/ 8 MLP blocks is our baseline with 8 blocks MLP action head.

Method	**All**	**PickCube**	**StackCube**	**SingleYCB**	**ClutterYCB**	**SingleEGAD**
Disc ActionHead	30.2%	41.0%	33.0%	22.0%	1.0%	54.0%
Diffusion Head w/ single token action chunks [69]	36.4%	68.0%	51.0%	26.0%	4.0%	33.0%
Diffusion Action Head	58.6%	86.0%	76.0%	37.0%	24.0%	70.0%
Diffusion Action Head w/ 8 MLP blocks	57.6%	**87.0%**	73.6%	58.3%	20.8%	53.1%
DiT Policy(ours)	**65.8%**	79.0%	**80.0%**	**62.0%**	**36.0%**	**72.0%**

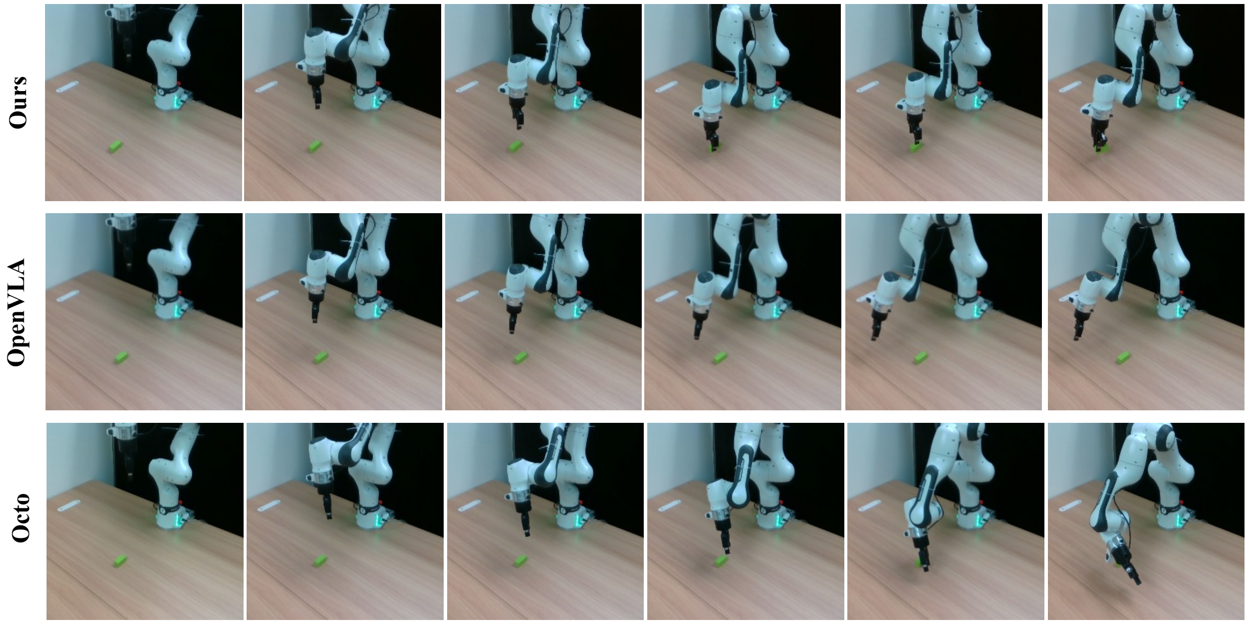


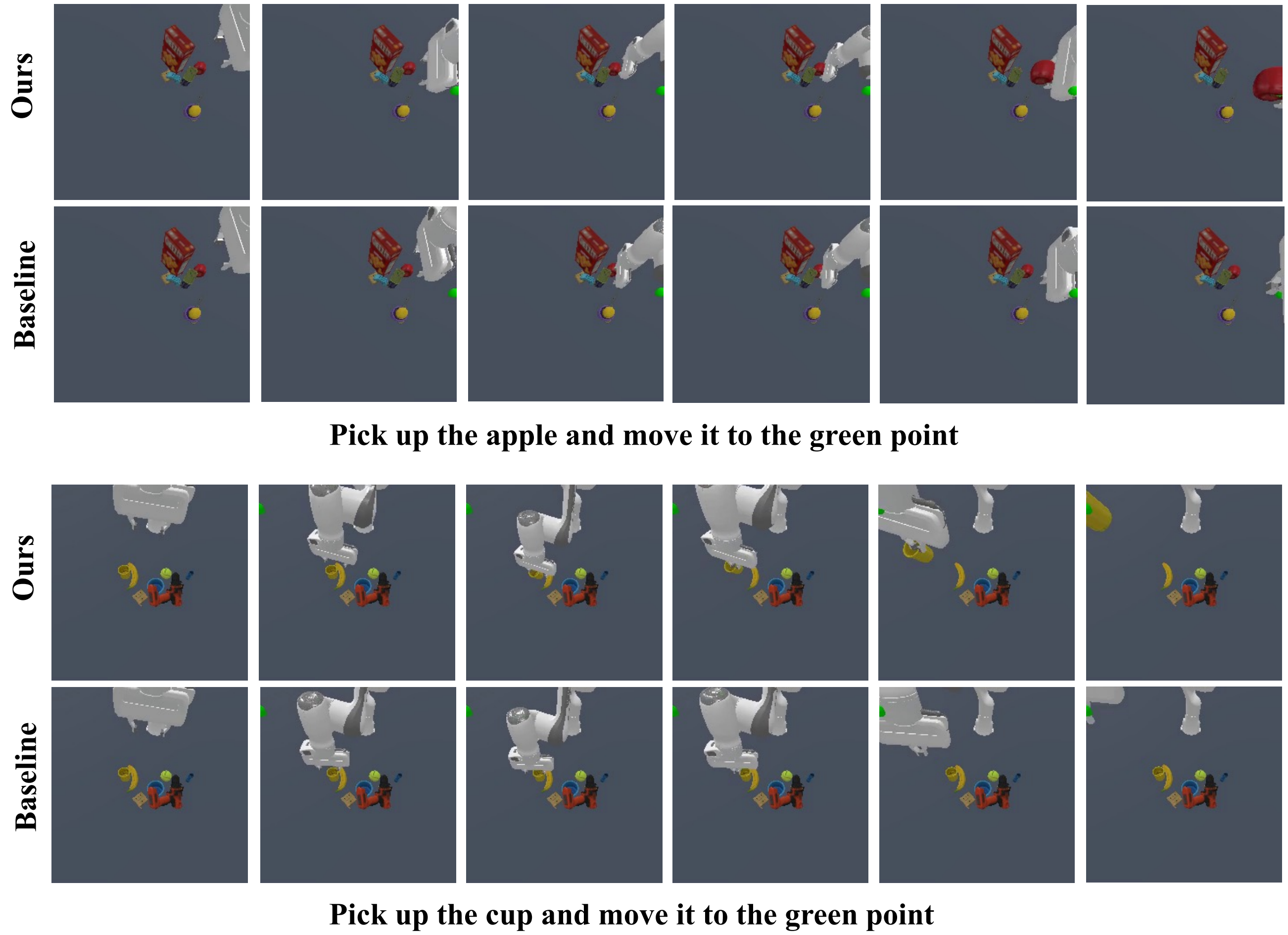
Figure 8. Visualization of zero-shot genearalizaton of different models. The first raw is Diffusion Transformer Policy(ours). The second raw of demonstration is OpenVLA [28], the third raw is Octo [69]. Our Model can complete the tasks successfully while both of the others fails.

Table 11. More action designs w/o pretraining on Calvin (ABC→D).

Methods	No. Instructions in a Row (1000 chains)					
Unet1D head [12]	76.8%	46.5%	28.8%	18.5%	10.0%	1.80
Transformer head [12]	75.8%	44.8%	26.5%	16.5%	8.0%	1.72
8 layers MLP head	69.8%	42.5%	26.3%	16.8%	11.0%	1.66
3 layers MLP head	75.5%	44.8%	25.0%	15.0%	7.5%	1.68
Single token act chunks	56.5%	18.3%	6.0%	2.8%	0.8%	0.84
Ours	**89.5%**	**63.3%**	**39.8%**	**27.3%**	**18.5%**	**2.38**

Table 12. The training dataset mixture

Fractal [6]	DobbE [61]	Droid [27]
16.15	1.94	13.69
Robo Set [29]	Viola [82]	Kuka [24]
2.99	1.30	17.47
BridgeV2 [70]	NYU Franka [13]	Furniture Bench [20]
21.86	1.14	6.73
StanfordHydra [1]	DLR EDAN [54]	BerkeleyFanuc [81]
6.11	0.08	1.07
Jaco [14]	LanguageTable [41]	toto [79]
0.67	6.01	2.78



Pick up the apple and move it to the green point

Pick up the cup and move it to the green point

Figure 9. Visualized comparison between Diffusion Transformer Policy and Diffusion Action Head baseline on Maniskill2 (PickClutterYCB). The first raw is Diffusion Tranformer Policy, while the second raw is the baseline method with Diffusion Action Head.

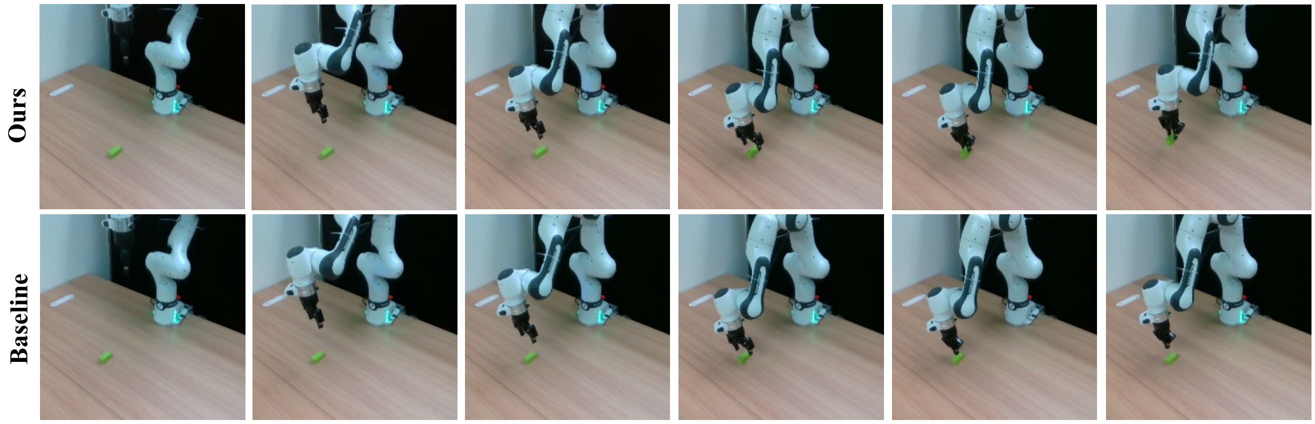


Figure 10. Visualized comparison between Diffusion Transformer Policy and Diffusion Action Head strategy on Real Franka Arm (Pick up the green block). The first raw is DiT Policy, while the second raw is the baseline method with Diffusion Action Head.