# 3D FlowMatch Actor: Unified 3D Policy for Single- and Dual-Arm Manipulation

Anonymous authors

*Abstract*— **We present 3D FlowMatch Actor (3DFA), a 3D policy architecture for robot manipulation that combines flow matching for trajectory prediction with 3D pretrained visual scene representations for learning from demonstration. 3DFA leverages 3D relative attention between action and visual tokens during action denoising, building on prior work in 3D diffusion-based single-arm policy learning. Through a combination of flow matching and targeted system-level and architectural optimizations, 3DFA achieves over 30× faster training and inference than previous 3D diffusion-based policies, without sacrificing performance. On the bimanual PerAct2 benchmark, it establishes a new state of the art, outperforming the next-best method by an absolute margin of 41.4%. In extensive real-world evaluations, it surpasses strong baselines with up to 1000× more parameters and significantly more pretraining. In unimanual settings, it sets a new state of the art on 74 RLBench tasks by directly predicting dense end-effector trajectories, eliminating the need for motion planning. Comprehensive ablation studies underscore the importance of our design choices for both policy effectiveness and efficiency.**

*Index Terms*— **Bimanual manipulation; Single-arm manipulation; Flow matching; Imitation learning**

## I. INTRODUCTION

Single-arm manipulation has achieved great success in handling long-horizon and high-precision tasks in cluttered environments [1], [2], [3]. However, the lack of coordination between multiple end-effectors largely constrains single-arm systems to simple pick-and-place tasks, making them inadequate for addressing the more complex and diverse manipulation challenges encountered in real-world daily tasks. To overcome these challenges, bimanual systems offer a promising alternative by enabling more dexterous and coordinated interactions with the environment.

Although bimanual setups improve the ability of a robot to perform more intricate and dexterous tasks, they also impose stricter demands on spatiotemporal precision. Both arms must operate in a tightly coordinated manner, executing actions in the correct temporal sequence and at precisely aligned spatial locations. This added complexity makes learning effective manipulation policies more difficult than in the single-arm case. Despite growing interest, existing approaches [4], [5], [6], [7], [8] still fall short of achieving robust generalization across a wide range of tasks.

In parallel, recent advances in single-arm manipulation have demonstrated the power of diffusion models in capturing multimodal behaviors [9], [10], [11], achieving high-precision action prediction through 3D scene understanding [12], [13], [14] and impressive generalization to various tasks and language instructions [15], [16]. A natural next step toward robust bimanual manipulation is to integrate these advances. In fact, we show that an adaptation of 3D Diffuser Actor (3DDA) [9] - a model that combines diffusion-based action generation with 3D scene representations - already establishes a new state of the art on the bimanual manipulation benchmark PerAct2 [7]. Next, we ask the question: what prevents 3DDA from being deployed in real-world bimanual manipulation scenarios? Our experimentation suggests two key bottlenecks: slow inference speed and long training time. For instance, on PerAct2, 3DDA requires approximately 21 days to train and operates at 0.5Hz during inference. The prolonged training time significantly limits the model's ability to adapt to new tasks, while the slow inference speed makes it unsuitable for real-time or dynamic task execution.

In light of these observations, we introduce 3D FlowMatch Actor (3DFA), a significantly more efficient extension of 3DDA that improves both training time and inference speed by an order of magnitude. On the PerAct2 bimanual manipulation benchmark [7], 3DFA reduces the training time from 21 days to 16 hours and increases the inference speed from 0.5Hz to 18.2Hz, without sacrificing performance. To enable faster inference, we replace the DDPM-based [17] formulation used in 3DDA with a Flow Matching approach [18], [19], reducing the number of denoising steps during inference from 100 to 5. To reduce training overhead, we implement a series of system-level optimizations for computational efficiency, such as faster token sampling, optimized data loading, efficient attention implementation and mixed-precision training. While none of these techniques is novel on its own, our contribution lies in carefully integrating them into a manipulation system.

3DFA achieves a new state of the art on the PerAct2 simulation benchmark, with a success rate of $85.1\%$. Furthermore, it outperforms strong baselines—including $\pi_0$ [15]—both in simulation and on a real-world 10-task benchmark we constructed using the bimanual ALOHA platform [20]. We conduct an extensive ablation study to break down the contributions of each design choice.

Notably, 3DFA is a general-purpose framework capable of predicting both sparse keyposes and dense end-effector trajectories, and is applicable to both unimanual and bimanual manipulation. On the 18 single-arm tasks of PerAct [12], 3DFA is trained to predict the next end-effector keypose and performs on par with 3DDA, while requiring significantly less training and inference time. Furthermore, on the 74-task benchmark of [21], 3DFA excels at jointly predicting the next keypose and the trajectory connecting it to the current pose in a single forward pass, outperforming the best baseline by $7.3\%$. These results highlight the versatility and efficiency of 3DFA as a framework for 3D manipulation.

In summary, our contributions are: (1) Adaptation of state-of-the-art single-arm 3D generative policies to bimanual manipulation, (2) Dramatic acceleration of training and inference time in 3D policies, (3) State-of-the-art bimanual manipulation results on PerAct2, with an absolute margin of 41.4% over $\pi_0$, (4) State-of-the-art bimanual manipulation results in the real world, outperforming foundational policies in a direct comparison, (5) State of the art on the HiveFormer unimanual 74-task benchmark, demonstrating planner-free trajectory prediction capabilities.

## II. Related Work

**Bimanual manipulation.** Bimanual manipulation is challenging due to the need for precise coordination between two arms. A key bottleneck is the difficulty of collecting high-quality bimanual demonstration data, which has historically constrained the scalability of approaches [4], [6]. Recent works [5], [20] have introduced more cost-effective pipelines and platforms for scaling real-world data collection. However, these methods primarily rely on RGB inputs and struggle to generalize across diverse tasks, object types or scene configurations. To address these limitations, several multi-task simulation benchmarks have been proposed. PerAct2 [7] extends the RLBench [2] benchmark to support multi-task bimanual manipulation, with expert demonstrations generated using sampling-based motion planners [22]. RoboTwin [23] introduces a digital twin framework to create diverse expert datasets and real-world-aligned evaluation environments.

The development of bimanual manipulation policies falls into two main categories. One line of research extends single-arm policy architectures to bimanual [8], [24]. The second line of research composes multiple single-arm policies into a unified bimanual policy [25], [26], [27]. Our work belongs to the first category and extends the action space of [9] to predict pose trajectories for both arms simultaneously.

**Diffusion and Flow Matching models in robotics.** Diffusion models have emerged as powerful tools in imitation learning [10], [11], [28] and offline RL [29], [30], [31]. DDPM [17] has been the most widely adopted diffusion algorithm to iteratively add and remove Gaussian noise to and from the samples, following conditional probability paths. More recently, Flow Matching [18], [19] has drawn attention in robot learning. Flow matching policies learn to predict the velocity field directly pointing toward the target action [15], [32], [33], [34], [35], [36], [37], [38], [39], [40], resulting in better sampling efficiency and lower computational cost than DDPM-based policies. 3DFA also incorporates Flow Matching, yielding a state-of-the-art bimanual manipulation policy with real-time inference capabilities.

## III. 3D FlowMatch Actor

The architecture of 3DFA is shown in Fig. 1. It is a robot policy for single- and dual-arm manipulation that generates 3D end-effector trajectories for one or more robot arms conditioned on the task instruction, proprioception history and scene visual information, through iterative denoising. 3DFA grounds visual and action tokens to 3D locations in a common coordinate frame, using camera calibration to transfer visual tokens from the camera frame to the robot's frame.

3DFA builds upon the state-of-the-art single-arm 3D diffusion policy, 3DDA [9], which we extend to also solve bimanual tasks. To accelerate inference and training, we replace the DDPM-based denoising with flow matching, adopt faster point sampling methods and attention implementations, and build a highly-efficient data loading pipeline.

We denote demonstrations as sequences of observations and actions $\{(\mathbf{o}_1, \mathbf{a}_1), (\mathbf{o}_2, \mathbf{a}_2), \ldots\}$, accompanied by a task language instruction $l$, where $\mathbf{o}_t$ denotes the visual observation and $\mathbf{a}_t$ the robot action at timestep $t$. Each observation $\mathbf{o}_t$ consists of one or more posed RGB-D images. Each action $\mathbf{a}_t$ is a single-arm end-effector pose and is decomposed into 3D location, rotation and binary (open/close) state: $\mathbf{a}_t = \{\mathbf{a}_t^{\text{loc}} \in \mathbb{R}^3, \mathbf{a}_t^{\text{rot}} \in \mathbb{R}^6, \mathbf{a}_t^{\text{open}} \in \{0, 1\}\}$. Let $\boldsymbol{\tau}_t = (\mathbf{a}_{t:t+T}^{\text{loc}}, \mathbf{a}_{t:t+T}^{\text{rot}})$ denote the trajectory of 3D locations and rotations at timestep $t$, with a temporal horizon $T$. At each timestep $t$, our model predicts one of more trajectories $\boldsymbol{\tau}_t$ and binary states $\mathbf{a}_{t:t+T}^{\text{open}}$. We first review Flow Matching in Sec. III-A and the 3DDA architecture in Sec. III-B. We then detail our extensions to support bimanual manipulation and the design choices to enhance efficiency in Sec. III-C.

### A. Flow Matching for Fast Action Generation

Diffusion models [17], [41] are powerful generative frameworks for modeling multimodal data distributions. They generate samples by iteratively removing Gaussian noise via a stochastic process defined by conditional probability transitions. In contrast, flow matching approaches [19], [18] generate data by solving an optimal transport problem between a source $\mu_0$ and a target distribution $\mu_1$.

In particular, we adopt Rectified Flow [18], an instantiation of flow matching that transforms a sample $X_0 \sim \mu_0$ into $X_1 \sim \mu_1$ by following straight paths in the sample space. This significantly reduces computation during inference while retaining the expressiveness needed for high-dimensional generation, making it especially well-suited for robotics, where real-time performance is critical. The rectified flow between $(X_0, X_1)$ defines a continuous, time-differentiable trajectory $\mathbf{Z} = Z_t : t \in [0, 1]$ that transports $X_0$ to $X_1$ and is governed by the ordinary differential equation (ODE):

$$dZ_t = v_t^X(Z_t)dt, \quad t \in [0, 1], \quad Z_0 = X_0, \tag{1}$$

where $v_t^X$ is a time-dependent velocity field. The optimal velocity field that minimizes the expected discrepancy from the straight-line path between $X_0$ and $X_1$ can be formally defined as: $\inf_v \int_0^1 \mathbb{E}[\|X_1 - X_0 - v(X_t, t)\|]dt$, where $X_t = (1-t)X_0 + tX_1$ denotes the linear interpolation at time $t$ [18]. The model is trained to approximate this velocity field by minimizing the loss [18]:

$$\mathcal{L}_\theta = \mathbb{E}_{t,X}[\|v_\theta(X_t, t) - v_t^X\|^2] \tag{2}$$

In our setting, the goal is to generate robot actions from noise. We define $\mu_0 \sim \mathcal{N}(0, I)$ and $\mu_1$ as the distribution over real actions. During inference, the model iteratively transforms

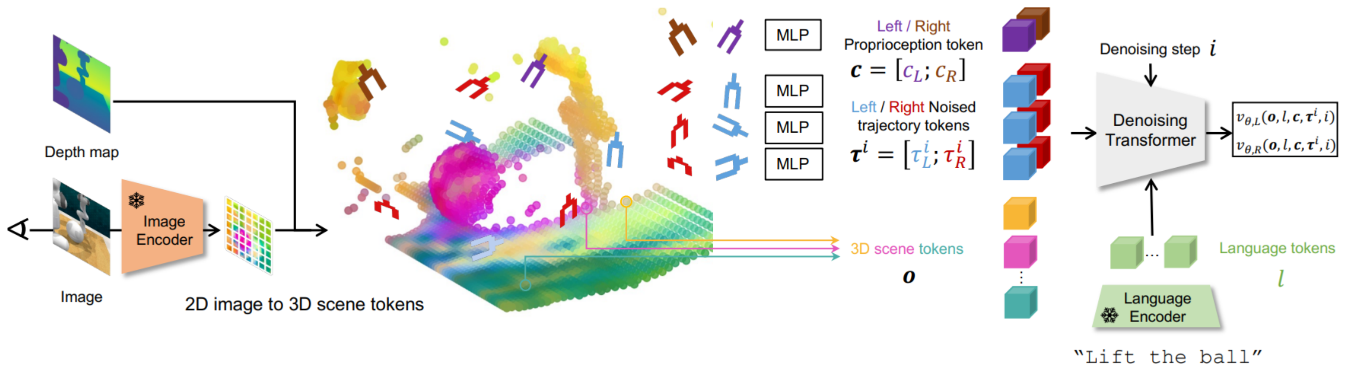

Depth map
Image
Image Encoder
2D image to 3D scene tokens
MLP
MLP
MLP
MLP
Left / Right Proprioception token
$c = [c_L; c_R]$
Left / Right Noised trajectory tokens
$\tau^i = [\tau_L^i; \tau_R^i]$
3D scene tokens
$o$
Denoising step $i$
Denoising Transformer
$v_{\theta,L}(o, l, c, \tau^i, i)$
$v_{\theta,R}(o, l, c, \tau^i, i)$
Language tokens
$l$
Language Encoder
"Lift the ball"

Fig. 1. 3D FlowMatch Actor model architecture. **3DFA** is a flow-matching policy built atop 3D Diffuser Actor [9]. It encodes the visual scene $\mathbf{o}$, robot proprioception for left and right arms $c_L$, $c_R$, and noised trajectories $\boldsymbol{\tau}_L^i$, $\boldsymbol{\tau}_R^i$ into 3D tokens. Given language tokens $l$, diffusion step $i$, and these 3D tokens, 3DFA predicts velocity fields $v_{\theta,L}$ and $v_{\theta,R}$ for the left and right arms, respectively.

the noise sample $X_0$ into a predicted action $X_1$ over $N$ steps with a fixed step size $\Delta t = \frac{1}{N}$: $X_{t+\Delta t} = X_t - \Delta t \cdot v_\theta(X_t, t)$.

## B. 3D Diffuser Actor

3DDA [9] is a 3D diffusion policy for manipulation that learns from demonstrations how to predict end-effector trajectories. By framing trajectory generation as a denoising process, 3DDA learns to iteratively refine noisy trajectory samples into clean end-effector motions, conditioning on multi-view RGB-D observations, language instructions and proprioception history. 3DDA conditions on 3D scene feature representations derived from posed cameras and sensed depth and uses DDPM-based diffusion to predict the noise component at each diffusion step. It distills 2D foundational features on point clouds to describe the scene and uses calibration information to transform the positional encodings of the 3D visual tokens into the robot's frame and to fuse them with action tokens that eventually decode the end-effector's translation and rotation at each future timestep.

To simplify notation, we denote $\boldsymbol{\tau}^i$ as the noisy trajectory estimate at diffusion step $i$ without specifying the trajectory timestep. The model conditions on the following inputs: (1) 3D scene tokens: 3DDA featurizes image views using a 2D image encoder and lifts each of the feature patches to 3D by calculating the average 3D location of each patch; (2) 3D proprioception tokens: 3DDA contextualizes a set of learnable embeddings with 3D scene tokens based on the proprioceptive location; (3) 3D trajectory tokens: 3DDA maps each noisy action $\mathbf{a}^i$ of trajectory $\boldsymbol{\tau}^i$ at diffusion step $i$ to a latent feature vector and lifts these feature vectors to 3D, based on the noisy location estimate of $\mathbf{a}^i$; (4) language tokens: language instructions are encoded to latent embeddings with a text encoder. 3DDA fuses all 3D tokens using relative 3D attentions, and additionally fuses language tokens using standard attention. As a last step, the refined trajectory tokens are passed through MLPs to predict the noise added to $\boldsymbol{\tau}^0$, as well as the end-effector opening.

## C. 3D FlowMatch Actor

We detail the core elements of 3DFA, as changes we have made over the model of [9].

*a) Extension to Bimanual Manipulation:* We first redefine the robot action in a bimanual form: $\mathbf{a}_{t,L}$ and $\mathbf{a}_{t,R}$ denote the robot action at timestep $t$, of the left and right robot arm respectively. Our goal is to predict the corresponding trajectory $\boldsymbol{\tau}_{t,L} = (\mathbf{a}^{\text{loc}}_{t:t+T,L}, \mathbf{a}^{\text{rot}}_{t:t+T,L})$ and $\boldsymbol{\tau}_{t,R} = (\mathbf{a}^{\text{loc}}_{t:t+T,R}, \mathbf{a}^{\text{rot}}_{t:t+T,r})$ of temporal horizon $T$ for both arms. To apply 3DFA on unimanual manipulation setups, we still use the unimanual trajectory definition: $\mathbf{a}_t$ and $\boldsymbol{\tau}_t = (\mathbf{a}^{\text{loc}}_{t:t+T}, \mathbf{a}^{\text{rot}}_{t:t+T})$. We follow the same 3D tokenization procedure to map (1) the noisy estimate of pose $\mathbf{a}_L^i$ of $\boldsymbol{\tau}_L^i$ and $\mathbf{a}_R^i$ of $\boldsymbol{\tau}_R^i$ at denoising step $i$, and (2) the left- and right-hand proprioceptive information $c_L$ and $c_R$, into 3D tokens. We use the same 3D Relative Denoising Transformer architecture to contextualize all these tokens and predict the translation and rotation noise and the end-effector opening for both arms.

*b) Flow Matching Action Prediction Objective:* We replace the DDPM-based diffusion method with *rectified flow*. The noisy left- and right-hand trajectory estimate $\boldsymbol{\tau}_L^i = (1-i)\epsilon_L + i\boldsymbol{\tau}_{t,L}^0$ and $\boldsymbol{\tau}_R^i = (1-i)\epsilon_R + i\boldsymbol{\tau}_{t,R}^0$ become the linear interpolation at denoising step $i$, where $\boldsymbol{\tau}_{t,L}^0$ and $\boldsymbol{\tau}_{t,R}^0$ denote the clean trajectory, and $\epsilon_L$ and $\epsilon_R$ denote the sampled noise for the left- and right-hand end effector. In particular, our model takes as input two-hand trajectory estimate tokens $\boldsymbol{\tau}^i = \{\boldsymbol{\tau}_L^i, \boldsymbol{\tau}_R^i\}$, proprioception tokens $\mathbf{c} = \{c_L, c_R\}$, language tokens $l$, and scene tokens $\mathbf{o}$; it outputs the left- and right-hand velocity field $v_{\theta,L}, v_{\theta,R}$ and gripper openness $f^{\text{open}}_{\theta,L}, f^{\text{open}}_{\theta,L}$, respectively. We ignore time step $t$ to simplify notations.

During training, we sample a time step $t$ uniformly, denoising step $i \sim \sigma(\mathcal{N}(0,I))$ from a logit-normal distribution and noise $\epsilon_L \sim \mathcal{N}(0,I), \epsilon_R \sim \mathcal{N}(0,I)$ from Gaussian distribution. We use the $L2$ loss to supervise the velocity field and binary cross-entropy (BCE) to supervise the end-effector openness. Omitting the notation of $t$, the objective reads:

$$\begin{aligned}\mathcal{L}_\theta &= \|\epsilon_L - \boldsymbol{\tau}_L^0 - v_{\theta,L}(\mathbf{o}, l, \mathbf{c}, \boldsymbol{\tau}^i, i)\|^2 \\ &+ \|\epsilon_R - \boldsymbol{\tau}_R^0 - v_{\theta,R}(\mathbf{o}, l, \mathbf{c}, \boldsymbol{\tau}^i, i)\|^2 \\ &+ \mathrm{BCE}(f^{\text{open}}_{\theta,L}(\mathbf{o}, l, \mathbf{c}, \boldsymbol{\tau}^i, i), \mathbf{a}^{\text{open}}_{1:T,L}) \\ &+ \mathrm{BCE}(f^{\text{open}}_{\theta,R}(\mathbf{o}, l, \mathbf{c}, \boldsymbol{\tau}^i, i), \mathbf{a}^{\text{open}}_{1:T,R}) \qquad (3)\end{aligned}$$

TABLE I

**EVALUATION ON PERACT2 BIMANUAL BENCHMARK.** 3DFA, ANYBIMANUAL AND $\pi_0$-KEYPOSE ARE MULTI-TASK POLICIES AND EVALUATE ONE CHECKPOINT ACROSS ALL TASKS, WHILE OTHERS ARE SINGLE-TASK POLICIES AND REPORT RESULTS FROM THE BEST CHECKPOINT PER TASK. **3DFA OUTPERFORMS ALL PRIOR ARTS BY A LARGE MARGIN.**

| | multi-task training | Avg. Success | push box | lift ball | push buttons | pick up plate | put item into drawer | put bottle into fridge | handover item | pick up laptop | straighten rope | sweep dust | lift tray | handover item (easy) | take tray out of oven |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ACT [5] | ✗ | 5.9 | 0 | 36 | 4 | 0 | 13 | 0 | 0 | 0 | 16 | 0 | 6 | 0 | 2 |
| RVT-LF [42], [7] | ✗ | 10.5 | 52 | 17 | 39 | 3 | 10 | 0 | 0 | 3 | 3 | 0 | 6 | 0 | 3 |
| PerAct-LF [12], [7] | ✗ | 17.5 | 57 | 40 | 10 | 2 | 27 | 0 | 0 | 11 | 21 | 28 | 14 | 9 | 8 |
| PerAct$^2$ [7] | ✗ | 16.8 | 6 | 50 | 47 | 4 | 10 | 3 | 11 | 12 | 24 | 0 | 1 | 41 | 9 |
| KStarDiffuser [43] | ✗ | - | 83 | 98.7 | - | - | - | - | - | 43.7 | - | 89 | - | 27 | - |
| PPI [44] | ✗ | - | **96.7** | 89.3 | - | - | 79.7 | - | - | 46.3 | - | 98.7 | 92 | 62.7 | - |
| AnyBimanual [25] | ✓ | 32 | 46 | 36 | 73 | 8 | - | 26 | 15 | 7 | 24 | 67 | 14 | 44 | 24 |
| $\pi_0$-keypose [15] | ✓ | 43.7 | 93 | 97 | 38 | 41 | 40 | 22 | 2 | 27 | 7 | 2 | 72 | 59 | 68 |
| 3DFA (ours) | ✓ | **85.1** | 92.7$_{\pm 0.47}$ | **99.7**$_{\pm 0.47}$ | **92.7**$_{\pm 1.89}$ | **69.7**$_{\pm 12.6}$ | **93.0**$_{\pm 2.83}$ | **89.3**$_{\pm 1.89}$ | **89.0**$_{\pm 7.12}$ | **74.0**$_{\pm 8.96}$ | **40.7**$_{\pm 1.89}$ | **99.3**$_{\pm 0.47}$ | **94.7**$_{\pm 0.47}$ | **96.0**$_{\pm 5.65}$ | **94.7**$_{\pm 1.89}$ |

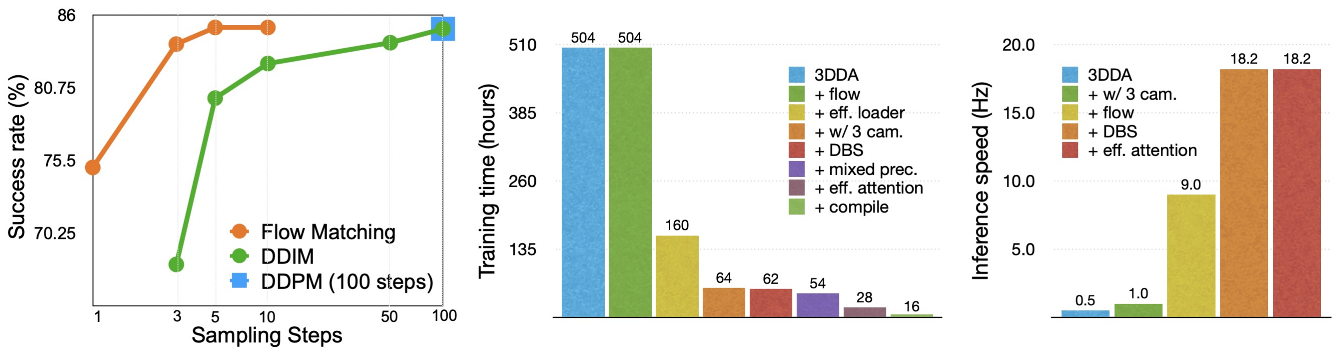


Fig. 2. **Ablation study** on PerAct2. **Left**: 3DFA's performance is stable even with as few as three denoising steps, contrary to the variants that use DDIM and DDPM. **Middle**: all design choices significantly contribute to lowering the training time from 504h to 16h (4 L40S GPUs). **Right**: Our contributions drastically improve inference speed from 0.5Hz to 18.2Hz (on one L40S GPU).

*c) Accelerating Dataloading:* We replace 3DDA's data loaders and optimize the keypose sampling during batching, data type conversion, and unprojection and augmentation operations. Specifically, we change the episodic loading to random keypose sampling. In more detail, the 3DDA codebase loads entire episodes, chunks them, and concatenates chunks from different episodes to form a batch. We, on the other hand, sample keyposes across random episodes. To efficiently do this, we used the Python library zarr to lazily load and access data indices across all episodes simultaneously. This offers the following advantages: i) we avoid loading whole episodes at once to only use a chunk, as this wastes time for data that is not used; ii) we ensure higher diversity in every batch; iii) we ensure a fixed batch size, contrary to 3DDA that concatenates chunks of possibly different sizes.

We handle data types to ensure faster batch collation. Specifically, we always load uint8 images and half-precision depth maps. This significantly speeds up batch formation, especially when large batch sizes are used. The data is converted to the desired type (float32) after being loaded to the GPU, where data-type conversions are much faster. In contrast, 3DDA loads and batches float32 tensors. We found that handling data types cuts down the loading time by half.

We move depth unprojection to point cloud and augmentations to GPU. This offers two advantages: i) loading single-channel depth is much faster than three-channel point clouds; ii) it allows for faster, batched GPU-optimized operations.

*d) Faster point sampling:* We adopt density-biased sampling (DBS) [45] to replace farthest point sampling (FPS) [46]. In more detail, 3DDA employs FPS in the feature space to sparsify the scene tokens. FPS maintains a set of candidate points and a set of sampled points. Then, it iteratively samples the candidate point with maximum average distance from all sampled points. We replace this with DBS, which first estimates the sparsity of a neighborhood around a point as the average distance of the $k = 8$ nearest neighbors of that point. Then, it promotes sampling in the sparser neighborhoods. A fast batched version of DBS can be implemented in pure PyTorch.

*e) Mixed-precision training:* We allow for autocasting operations to half precision when possible. This reduces the memory footprint of 3DFA and allows for larger batch sizes.

*f) Efficient attention implementation:* We use a modern PyTorch implementation of attention that runs optimized C++ code under the hood and is faster by an order of magnitude when combined with large batch sizes and half precision.

*g) CUDA graph compilation:* In modern PyTorch versions, a model can be compiled as a graph of optimized non-dynamic operations. Making 3DFA compilable required refactoring changes over 3DDA, such as removal of logic branches, CPU operations, in-graph language tokenization and custom kernel operations such as FPS.

## IV. EXPERIMENTS

We evaluate 3DFA on learning manipulation behaviors from demonstration in simulation and the real world: PerAct2 bimanual manipulation benchmark (Sec. IV-A), PerAct (Sec. IV-B) and HiveFormer (Sec. IV-C) unimanual manipulation benchmarks, and a suite of 10 real-world bimanual tasks using the Aloha robot platform (Sec. IV-D).

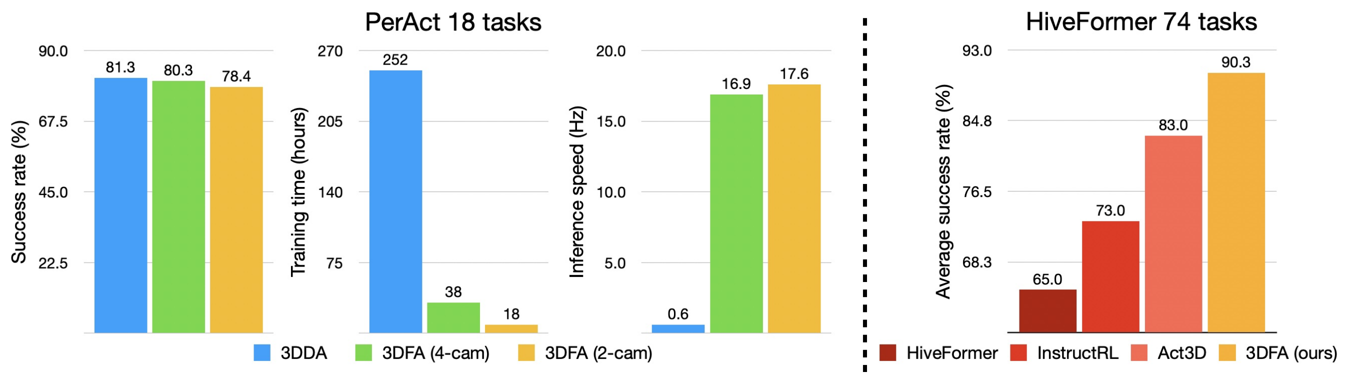


Fig. 3. **Single-arm manipulation results. Left**: On the PerAct 18-task benchmark, 3DFA matches 3DDA's performance while reducing training time by over 6× and achieving 28× faster inference. **Right**: 3DFA achieves a new state-of-the-art on the 74 tasks of HiveFormer.

## A. *Evaluation on the PerAct2 Bimanual Manipulation Benchmark*

PerAct2 [7] is a simulation benchmark for bimanual manipulation using two Franka Emika Panda robots. It includes 13 tasks, each with 1 to 5 variations that involve changes in object pose, appearance or semantics . For each task, the dataset provides 100 demonstrations for training and 100 episodes for evaluation. Methods are trained to predict the next end-effector keypose [7], which is then passed to an RRT-based motion planner to generate a feasible joint trajectory from the current configuration to the predicted pose. PerAct2 supports five calibrated RGB-D cameras—front, left wrist, right wrist, left shoulder and right shoulder—capturing images at a resolution of $256 \times 256$. We find that using only the front, left wrist and right wrist is sufficient for all tasks. Task success rate is used as the primary evaluation metric.

*a) Baselines:* We compare our method against several strong baselines, including: (1) ACT [5], a 2D transformer architecture that is trained as a conditional VAE to predict a sequence of actions; (2) RVT-LF [42], [7], that unprojects 2D views to form a point cloud, renders virtual views and feeds them to a transformer to predict the 3D actions for each arm in sequence; (3) PerAct-LF [12], [7], that voxelizes the 3D space and uses a Perceiver [47] architecture to predict the 3D actions for each arm in sequence; (4) PerAct$^2$ [7], which shares the same architecture as PerAct-LF but predicts the actions for the two arms jointly; (5) AnyBimanual [25], which combines and adapts two pre-trained single-arm PerAct [12] policies; (6) 3D Diffusion Policy (DP3) [48], which encodes 3D scenes with a point cloud encoder and uses a diffusion UNet [49] to predict the 3D actions; (7) KStar Diffuser [43], a diffusion graph convolutional network that regularizes end-effector pose prediction with predicting body joint angles; (8) PPI [44], a 3D diffusion policy that regularizes action prediction by tracking points sampled from objects of interest; (9) $\pi_0$[15]. Notably, $\pi_0$ is a 2D robot policy pretrained on 10,000 hours of robot demonstration data and capable of both unimanual and bimanual manipulation. It takes visual input from three camera views (front and wrists) and outputs future joint angle trajectories. We adapt $\pi_0$ to predict 3D end-effector keyposes, which is the standard output format used across all policies

TABLE II

EVALUATION ON 8 RLBENCH TASKS FROM [14] THAT REQUIRE CONTINUOUS INTERACTION WITH THE ENVIRONMENT. 3DFA PREDICTS DENSE TRAJECTORIES AND OUTPERFORMS ALL BASELINES BY A LARGE MARGIN.

| | Avg. Success | Num. stages | unplug charger | close door | open box | open fridge | frame off hanger | open oven | books in shelf | shoes in box |
|---|---|---|---|---|---|---|---|---|---|---|
| DP3 [48] | 28.5 | 1 | 33.3 | 76.0 | 98.3 | 4.3 | 12.3 | 0.3 | 3.7 | 0.0 |
| PointFlowMatch [37] | 67.8 | 1 | 83.6 | 68.3 | 99.4 | 31.9 | 38.6 | 75.9 | 68.8 | 76.0 |
| ChainedDiffuser [14] | 84.5 | 2 | 95.0 | 76.0 | 96.0 | 68.0 | 85.0 | 86.0 | 92.0 | **78.0** |
| 3DFA (ours) | **91.3** | 1 | **99.0** | **96.0** | **100.0** | **70.0** | **96.0** | **99.0** | **99.0** | 71.0 |

in this setting and significantly improves its performance. We call this variant $\pi_0$-keypose. Our method, along with $\pi_0$-keypose and AnyBimanual, uses *multitask training* and evaluates performance using the final checkpoint, whereas other baselines train separate models per task and report results from the best intermediate checkpoint for each task. Second, several baselines—including AnyBimanual, DP3, PPI, and KStarDiffuser—are evaluated on *only a subset* of the 13 benchmark tasks.

*b) Results:* We show quantitative results in Tab. I. 3DFA largely outperforms all baselines, with an absolute improvement of 68.3% over PerAct$^2$. When isolating the same 5 tasks in which both PPI and KStarDiffuser report results, 3DFA achieves 92.3%, outperforming them by 13.6% and 24% absolute respectively. Notably, our method with 3.8M parameters outperforms $\pi_0$ which has nearly 1000 times more parameters. We show that, although large-scale pretraining is useful, explicitly incorporating 3D information into the model is a strong inductive bias. We also test 3DDA on this benchmark, which also achieves an average success rate of 85%.

*c) Ablations:* We compare different denoising methods and quantify the effect of each contribution on training time and control frequency (Fig. 2). Key observations:

- **Denoising method impact**: DDPM is the least flexible, requiring 100 denoising steps to achieve 85.0% SR. DDIM [50] reduces steps but suffers a sharp performance drop. Flow matching is the most robust, maintaining full 85.1% performance even with 5 steps, 83.9% with 3 steps, and 75.2% with just 1 step.
- **Training time reduction**: Starting from the original

3DDA (adapted for bimanual manipulation), replacing its data-loading scheme with ours cuts training time by two-thirds. Additional optimizations cumulatively yield a 10× speedup, for a **total 30× training speedup**.

- **Control frequency gains**: Switching from DDPM to flow matching reduces denoising steps by 20× but raises control frequency only 9×, as scene encoding dominates the forward pass. Replacing FPS with DBS halves this cost, doubling control frequency. The attention implementation has negligible effect at test time, as it is tuned for large batch sizes.

## B. Evaluation on the PerAct Unimanual Manipulation Benchmark

We evaluate our method on the 18-task PerAct benchmark [12] to enable a direct comparison with 3DDA [9] in terms of both performance and efficiency. All policies are trained in a multi-task setting using 100 demonstrations per task and evaluated over 25 episodes per task. The setup includes four calibrated RGB-D cameras: front, wrist, left shoulder and right shoulder.

We compare two variants of 3DFA against 3DDA in Fig. 3, with all models trained to predict keyposes. When using all four cameras, 3DFA achieves performance on par with 3DDA while offering a 28× faster inference speed and requiring 6× less training time. With only two cameras (front and wrist), 3DFA further reduces computational cost—14× less training time and 30× faster inference—with minor performance drop.

## C. Evaluation on the HiveFormer Unimanual Manipulation Benchmark

We evaluate 3DFA on the 74-task HiveFormer benchmark [21] to showcase its ability to predict dense end-effector trajectories rather than sparse keyposes. This capability enables planner-free execution, which is critical for tasks requiring continuous interaction with objects and the environment. For instance, when opening a fridge, the end-effector must follow the door's rotational arc to respect its mechanical limits—a constraint that an RRT planner does not account for.

All policies are trained on one task at a time with 100 demonstrations and evaluated on 100 test episodes. We report results across the full 74-task benchmark and also highlight performance on a subset of 8 challenging tasks [14], [37] that demand continuous interaction and cannot be solved by simply predicting keyposes.

*a) Baselines:* We compare against: (1) keypose-prediction models HiveFormer [21], InstructRL [52] and Act3D [13], which use three cameras; (2) close-loop trajectory-prediction models PointFlowMatch [37] and DP3 [48], which use five cameras; (3) ChainedDiffuser [14], a two-stage policy consisting of a keypose predictor and a keypose-conditioned trajectory predictor, using three cameras.

3DFA is trained to jointly predict the next end-effector keypose and the dense trajectory until the next keypose in a non-hierarchical, single-forward-pass fashion. It relies on two camera observations, front and wrist, to maintain consistency with the rest of our experiments.

We present results across all 74 tasks in Fig. 3. 3DFA sets a new state of the art with a success rate of 90.3%, surpassing Act3D by 7.3%. On eight particularly challenging tasks (Tab. II), our method outperforms the two-stage ChainedDiffuser by 6.8%, demonstrating its strong capability for continuous trajectory prediction.

## D. Evaluation in the Real World

We construct a challenging real-world multi-task manipulation benchmark using Mobile Aloha [53], a dual-arm mobile manipulator equipped with a front-facing ZEDX RGB-D camera and two wrist-mounted RealSense D405 RGB-D sensors. Our benchmark (Fig. 4) comprises 10 tasks that demand precise and coordinated two-handed manipulation, examples include *open marker*, *close ziploc*, and *insert battery*, surpassing the complexity of PerAct2 tasks. We collect 40 demonstrations per task, recording visual observations and joint actions at 5 Hz. All models are trained or fine-tuned using the same demonstration set and evaluated on 20 episodes per task.

*a) Baselines:* We compare against two strong baselines: (1) $\pi_0$ [15], a generalist 2D manipulation policy, and (2) iDP3 [51], a variant of DP3 adapted for dual-arm humanoid systems with architectural enhancements. All models, including ours, are trained for closed-loop trajectory prediction without intermediate keypose supervision. Following their original designs, both baselines directly predict joint angles for the robot arms, whereas 3DFA predicts 3D end-effector poses, which are then converted to joint commands via inverse kinematics. To mimic human operation, all models output joint values for the Aloha leader arms, with the follower arms mirroring the motion. 3DFA and iDP3 are 3D policies that require accurate depth sensing; due to high noise from the wrist-mounted depth sensors, we exclude wrist camera inputs for both. In contrast, the 2D $\pi_0$ policy is depth-independent and utilizes all three camera views. All image inputs are downsampled to a resolution of $256 \times 256$.

*b) Results:* We show the quantitative results in Tab. III. 3DFA outperforms $\pi_0$ and iDP3 on most tasks. We find that $\pi_0$ is sensitive to occlusion and cannot locate and reach the object if it is not visible in the wrist observations. We also find that iDP3 struggles to predict precise end-effector poses, as the robot often approaches but fails to grasp the object.

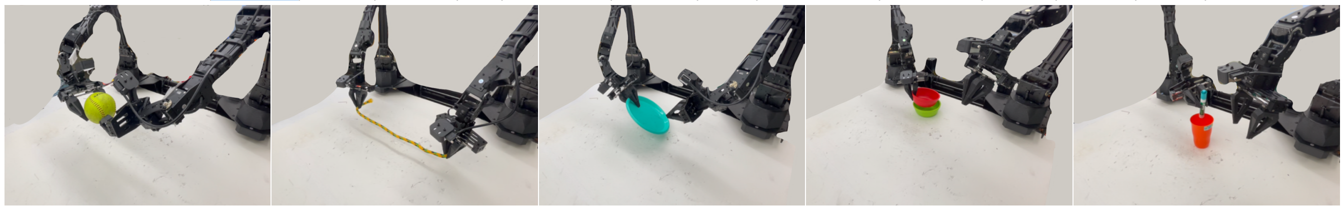
## E. Limitations and Future Directions

While 3DFA significantly reduces training and inference costs, it retains a key limitation: as a 3D policy, it depends on accurate depth sensing and camera calibration—resources often unavailable in large-scale, real-world imitation learning datasets. In particular, wrist-mounted cameras present the greatest calibration challenges. We are exploring approaches to relax these depth and calibration requirements, including architectures capable of jointly processing 2D and 3D observations [54], as well as leveraging recent advances in

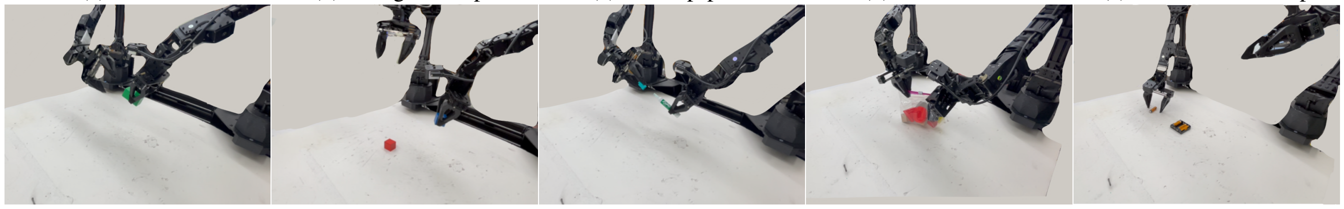
TABLE III

**Evaluation in the real world.** 3DFA and $\pi_0$ adopt multi-task learning settings, while iDP3 uses a single-task learning setup. **3DFA outperforms $\pi_0$ and iDP3 on all tasks by a large margin.**

| | Avg. Success | Inf. speed | Params in M | lift ball | straighten rope | pick up plate | stack bowl | put marker into bowl | handover block | stack blocks | open marker | close ziploc | insert battery |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\pi_0$ [15] | 32.5 | 100ms | 3238 | 85 | 75 | 40 | 20 | 20 | 15 | 35 | 20 | 10 | 5 |
| iDP3$^{\dagger}$ [51] | 24.5 | 420ms | 68.8 | 45 | 35 | 30 | 40 | 35 | 25 | 15 | 10 | 10 | 0 |
| 3DFA (ours) | **53.5** | 54ms | 3.8 | **85** | **75** | **80** | **80** | **70** | **60** | **45** | **25** | **10** | **5** |



(a) Lift ball (b) Straighten rope (c) Pick up plate (d) Stack bowls (e) Put marker in cup



(f) Hand over block (g) Stack blocks (h) Open marker (i) Close ziploc (j) Insert battery

Fig. 4. **Real-world benchmark.** Our real-world benchmark consists of 10 bimanual tasks, divided into 5 easy tasks (top) and 5 difficult tasks (bottom row).

metric depth estimation and automatic calibration from the broader computer vision community.

The compact parameterization of 3DFA, while contributing to its efficiency, also makes it susceptible to certain errors. For instance, on PerAct, the model may struggle with unseen variations encountered at test time. In our real-world benchmark, it has difficulty with tasks demanding fine force control or extreme precision. Scaling up the model's capacity and incorporating strong vision-language models for richer visual and language understanding are promising directions for addressing these challenges.

## V. Conclusion

We introduced 3D FlowMatch Actor (3DFA), a fast and versatile 3D manipulation policy that combines flow matching with pretrained 3D scene representations. Through targeted architectural and system-level optimizations, 3DFA achieves over 30x faster training and inference than prior 3D diffusion-based policies, while setting a new state of the art on both bimanual (PerAct2) and unimanual (RLBench-74) benchmarks. It delivers real-time performance, scales to real-world tasks, and removes the need for motion planning via direct dense trajectory prediction. These results position 3DFA as an efficient and general-purpose framework for unimanual and bimanual robot manipulation.

## References

[1] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone, "Libero: Benchmarking knowledge transfer for lifelong robot learning," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[2] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison, "Rlbench: The robot learning benchmark & learning environment," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3019–3026, 2020.

[3] X. Li, K. Hsu, J. Gu, K. Pertsch, O. Mees, H. R. Walke, C. Fu, I. Lunawat, I. Sieh, S. Kirmani *et al.*, "Evaluating real-world robot manipulation policies in simulation," *arXiv preprint arXiv:2405.05941*, 2024.

[4] J. Grannen, Y. Wu, S. Belkhale, and D. Sadigh, "Learning bimanual scooping policies for food acquisition," in *Conference on Robot Learning*, 2022.

[5] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," *RSS*, 2023.

[6] J. Grannen, Y. Wu, B. Vu, and D. Sadigh, "Stabilize to act: Learning to coordinate for bimanual manipulation," *CoRL*, 2023.

[7] M. Grotz, M. Shridhar, T. Asfour, and D. Fox, "Peract2: A perceiver actor framework for bimanual manipulation tasks," *arXiv preprint arXiv:2407.00278*, 2024.

[8] I.-C. A. Liu, S. He, D. Seita, and G. Sukhatme, "Voxact-b: Voxel-based acting and stabilizing policy for bimanual manipulation," *CoRL*, 2024.

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

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

[11] M. Reuss, M. Li, X. Jia, and R. Lioutikov, "Goal-conditioned imitation learning using score-based diffusion policies," *arXiv preprint arXiv:2304.02532*, 2023.

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

[13] T. Gervet, Z. Xian, N. Gkanatsios, and K. Fragkiadaki, "Act3d: 3d feature field transformers for multi-task robotic manipulation," *CoRL*, 2023.

[14] Z. Xian, N. Gkanatsios, T. Gervet, T.-W. Ke, and K. Fragkiadaki, "Chaineddiffuser: Unifying trajectory diffusion and keypose prediction for robotic manipulation," in *Conference on Robot Learning*. PMLR, 2023, pp. 2323–2339.

[15] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai,

L. Groom, K. Hausman, B. Ichter *et al.*, "π0: A vision-language-action flow model for general robot control, 2024," *arXiv preprint arXiv:2410.24164*, 2024.

[16] D. Qu, H. Song, Q. Chen, Y. Yao, X. Ye, Y. Ding, Z. Wang, J. Gu, B. Zhao, D. Wang *et al.*, "Spatialvla: Exploring spatial representations for visual-language-action model," *arXiv preprint arXiv:2501.15830*, 2025.

[17] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *CoRR*, vol. abs/2006.11239, 2020. [Online]. Available: https://arxiv.org/abs/2006.11239

[18] X. Liu, C. Gong, and Q. Liu, "Flow straight and fast: Learning to generate and transfer data with rectified flow," *arXiv preprint arXiv:2209.03003*, 2022.

[19] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," *arXiv preprint arXiv:2210.02747*, 2022.

[20] Z. Fu, T. Z. Zhao, and C. Finn, "Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation," 2024.

[21] P.-L. Guhur, S. Chen, R. G. Pinel, M. Tapaswi, I. Laptev, and C. Schmid, "Instruction-driven history-aware policies for robotic manipulations," in *6th Annual Conference on Robot Learning*, 2022. [Online]. Available: https://openreview.net/forum?id=h0Yb0U_-Tki

[22] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 2. IEEE, 2000, pp. 995–1001.

[23] Y. Mu, T. Chen, Z. Chen, S. Peng, Z. Lan, Z. Gao, Z. Liang, Q. Yu, Y. Zou, M. Xu, L. Lin, Z. Xie, M. Ding, and P. Luo, "Robotwin: Dual-arm robot benchmark with generative digital twins," 2025. [Online]. Available: https://arxiv.org/abs/2504.13059

[24] S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu, "Rdt-1b: a diffusion foundation model for bimanual manipulation," *arXiv preprint arXiv:2410.07864*, 2024.

[25] G. Lu, T. Yu, H. Deng, S. S. Chen, Y. Tang, and Z. Wang, "Anybimanual: Transferring unimanual policy for general bimanual manipulation," *arXiv preprint arXiv:2412.06779*, 2024.

[26] T. Motoda, R. Hanai, R. Nakajo, M. Murooka, F. Erich, and Y. Domae, "Learning bimanual manipulation via action chunking and inter-arm coordination with transformers," *arXiv preprint arXiv:2503.13916*, 2025.

[27] J.-J. Jiang, X.-M. Wu, Y.-X. He, L.-A. Zeng, Y.-L. Wei, D. Zhang, and W.-S. Zheng, "Rethinking bimanual robotic manipulation: Learning with decoupled interaction framework," *arXiv preprint arXiv:2503.09186*, 2025.

[28] N. Gkanatsios, A. Jain, Z. Xian, Y. Zhang, C. Atkeson, and K. Fragkiadaki, "Energy-based models as zero-shot planners for compositional scene rearrangement," *arXiv preprint arXiv:2304.14391*, 2023.

[29] H. Chen, C. Lu, C. Ying, H. Su, and J. Zhu, "Offline reinforcement learning via high-fidelity generative behavior modeling," 2023.

[30] B. Yang, H. Su, N. Gkanatsios, T.-W. Ke, A. Jain, J. Schneider, and K. Fragkiadaki, "Diffusion-es: Gradient-free planning with diffusion for autonomous driving and zero-shot instruction following," *ArXiv*, vol. abs/2402.06559, 2024.

[31] P. Hansen-Estruch, I. Kostrikov, M. Janner, J. G. Kuba, and S. Levine, "Idql: Implicit q-learning as an actor-critic method with diffusion policies," *arXiv preprint arXiv:2304.10573*, 2023.

[32] N. Funk, J. Urain, J. Carvalho, V. Prasad, G. Chalvatzaki, and J. Peters, "Actionflow: Equivariant, accurate, and efficient policies with spatially symmetric flow matching," *arXiv preprint arXiv:2409.04576*, 2024.

[33] M. Braun, N. Jaquier, L. Rozo, and T. Asfour, "Riemannian flow matching policy for robot motion learning," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 5144–5151.

[34] F. Zhang and M. Gienger, "Affordance-based robot manipulation with flow matching," *arXiv preprint arXiv:2409.01083*, 2024.

[35] H. Ding, N. Jaquier, J. Peters, and L. Rozo, "Fast and robust visuomotor riemannian flow matching policy," *arXiv preprint arXiv:2412.10855*, 2024.

[36] Q. Zhang, Z. Liu, H. Fan, G. Liu, B. Zeng, and S. Liu, "Flowpolicy: Enabling fast and robust 3d flow-based policy via consistency flow matching for robot manipulation," *arXiv preprint arXiv:2412.04987*, 2024.

[37] E. Chisari, N. Heppert, M. Argus, T. Welschehold, T. Brox, and A. Valada, "Learning robotic manipulation policies from point clouds with conditional flow matching," *arXiv preprint arXiv:2409.07343*, 2024.

[38] S. Wang, L. Wang, S. Zhou, J. Tian, J. Li, H. Sun, and W. Tang, "Flowram: Grounding flow matching policy with region-aware mamba framework for robotic manipulation," *Conference on Computer Vision and Pattern Recognition*, 2025.

[39] J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, S. Huang *et al.*, "Gr00t n1: An open foundation model for generalist humanoid robots," *arXiv preprint arXiv:2503.14734*, 2025.

[40] M. Reuss, H. Zhou, M. Rühle, Ö. E. Yağmurlu, F. Otto, and R. Lioutikov, "Flower: Democratizing generalist robot policies with efficient vision-language-action flow policies," in *7th Robot Learning Workshop: Towards Robots with Human-Level Abilities*, 2025.

[41] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," *arXiv preprint arXiv:2011.13456*, 2020.

[42] A. Goyal, J. Xu, Y. Guo, V. Blukis, Y.-W. Chao, and D. Fox, "Rvt: Robotic view transformer for 3d object manipulation," *arXiv preprint arXiv:2306.14896*, 2023.

[43] Q. Lv, H. Li, X. Deng, R. Shao, Y. Li, J. Hao, L. Gao, M. Y. Wang, and L. Nie, "Spatial-temporal graph diffusion policy with kinematic modeling for bimanual robotic manipulation," *arXiv preprint arXiv:2503.10743*, 2025.

[44] Y. Yang, Z. Cai, Y. Tian, J. Zeng, and J. Pang, "Gripper keypose and object pointflow as interfaces for bimanual robotic manipulation," *RSS*, 2025.

[45] C. R. Palmer and C. Faloutsos, "Density biased sampling: an improved method for data mining and clustering," in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '00. New York, NY, USA: Association for Computing Machinery, 2000, p. 82–92. [Online]. Available: https://doi.org/10.1145/342009.335384

[46] Y. Eldar, M. Lindenbaum, M. Porat, and Y. Zeevi, "The farthest point strategy for progressive image sampling," *IEEE Transactions on Image Processing*, vol. 6, no. 9, pp. 1305–1315, 1997.

[47] A. Jaegle, F. Gimeno, A. Brock, A. Zisserman, O. Vinyals, and J. Carreira, "Perceiver: General perception with iterative attention," 2021.

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

[49] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer, 2015, pp. 234–241.

[50] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," 2022.

[51] Y. Ze, Z. Chen, W. Wang, T. Chen, X. He, Y. Yuan, X. B. Peng, and J. Wu, "Generalizable humanoid manipulation with improved 3d diffusion policies," *arXiv preprint arXiv:2410.10803*, 2024.

[52] H. Liu, L. Lee, K. Lee, and P. Abbeel, "Instruction-following agents with jointly pre-trained vision-language models," *arXiv preprint arXiv:2210.13431*, 2022.

[53] Z. Fu, T. Z. Zhao, and C. Finn, "Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation," *arXiv preprint arXiv:2401.02117*, 2024.

[54] A. Jain, P. Katara, N. Gkanatsios, A. W. Harley, G. Sarch, K. Aggarwal, V. Chaudhary, and K. Fragkiadaki, "Odin: A single model for 2d and 3d segmentation," 2024. [Online]. Available: https://arxiv.org/abs/2401.02416