

# Motion Before Action: Diffusing Object Motion as Manipulation Condition

Yue Su<sup>\*1,2</sup>, Xinyu Zhan<sup>\*1</sup>, Hongjie Fang<sup>1</sup>, Yong-Lu Li<sup>1</sup>, Cewu Lu<sup>1</sup>, and Lixin Yang<sup>1†</sup>

**Abstract**—Inferring object motion representations from observations enhances the performance of robotic manipulation tasks. This paper introduces a new paradigm for robot imitation learning that generates action sequences by reasoning about object motion from visual observations. We propose MBA, a novel module that employs two cascaded diffusion processes for object motion generation and robot action generation under object motion guidance. MBA first predicts the future pose sequence of the object based on observations, and then uses this sequence as a condition to guide robot action generation. Designed as a plug-and-play component, MBA can be flexibly integrated into existing robotic manipulation policies with diffusion action heads. Extensive experiments in both simulated and real-world environments demonstrate that our approach substantially improves the performance of existing policies across a wide range of manipulation tasks. Project page: <https://selen-suyue.github.io/MBApage/>

## I. INTRODUCTION

Physiological research shows that humans process complex object motion information in their environment to support effective action execution [6]. This motion analysis enables an understanding of object dynamics, which in turn guides human actions such as reaching, grasping, and maneuvering around obstacles.

In contrast to the biological approach, most existing robot policies [8, 28, 42, 46, 53, 56] are predominantly guided by observation, employing feature encoders and adopting generative approaches to predict actions. While effective, it often results in an overreliance on environmental cues, with the model focusing on memorizing observation features rather than reasoning about object motion patterns, as humans do. Consequently, when encountering extensive pose shifts in real-world objects or actions, many policies often struggle to generalize effectively [22, 28], which can limit their practical performance.

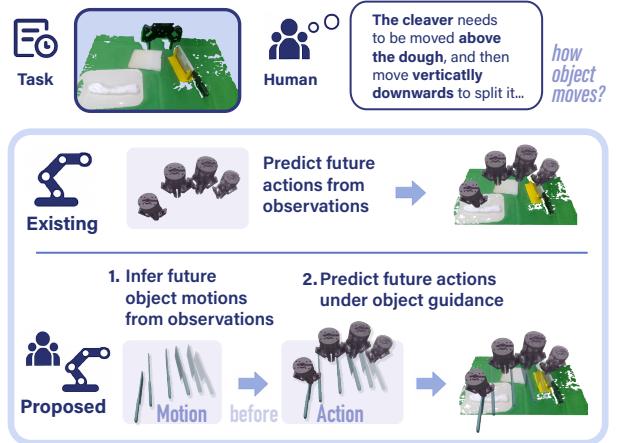
To address these challenges and improve execution capabilities, we aim to equip the robot with human-like reasoning skills by *inferring future object motion from observations*, and then *predicting future actions under the object motion guidance*. By achieving these objectives, the robot can derive intrinsic information (such as poses and motions) from the scene, allowing the policy to map observations to actions in a way that aligns more closely with human reasoning, *i.e.*, reasoning about object motion rather than simply memorizing actions [20].

<sup>\*</sup>Equal Contribution.

<sup>1</sup>Shanghai Jiao Tong University. <sup>2</sup>Xidian University (this work was done while Y. Su is a research intern at Shanghai Jiao Tong University).

<sup>†</sup>Lixin Yang is the corresponding author. He is with the School of Artificial Intelligence, Shanghai Jiao Tong University, Shanghai, 200230, China

Author emails: s3702681@gmail.com, {kelvin34501, galaxies, yonglu.li, lucewu, siriusyang}@sjtu.edu.cn.



**Fig. 1: Understanding object motion before action leads to better manipulation.** Unlike existing methods that predict actions directly from observations, our approach first infers future object motions, enabling more accurate and goal-driven action prediction.

In this work, we introduce **Motion Before Action** (referred to as **MBA**), a novel module that equips existing robotic manipulation policies with such reasoning skills. We observe that representing the object motion using 6D pose aligns with the robot’s end-effector pose in the same format, making the actions compatible with this representation. Consequently, they exhibit mathematical consistency. During task execution, at each time step, the robot pose, object pose, and action are in proximity within the same space, with their kinematic relationships following a learnable pattern, demonstrating physical consistency [48]. These consistencies suggest that the object poses and the robot poses and actions share similar distributions. The ability of probabilistic models, exemplified by diffusion models, to learn such similar distributions is transferable, as indicated in [17, 34]. Therefore, we propose that object poses can be generated by the diffusion process in the same way as actions. These object poses can also effectively guide the action denoising process as conditions, just as robot poses do.

Building on this foundation, we design MBA as a cascade of two diffusion modules. MBA seamlessly integrates with any existing diffusion-based policy. The observation features encoded by the given policy serve as conditional inputs to the first diffusion module, which predicts future object pose sequences. These predicted pose sequences, combined with the initial observations, are then fed as joint conditions to the diffusion action head, effectively guiding the policy’s action generation.

We conduct comparative experiments with MBA on three 2D and 3D robotic manipulation policies with diffusion ac-

tion heads [8, 42, 53], demonstrating substantial performance improvements across various tasks. These tasks, comprising 57 tasks from 3 simulation benchmarks and 4 real-world tasks, involve articulated object manipulation, soft and rigid body manipulation, tool use, non-tool use, and diverse action patterns. Results show that MBA consistently enhances the performance of such policies in both simulated and real-world environments.

In this paper, we make the following key contributions:

- We propose a novel imitation learning paradigm for robotic manipulation that allows robots to extract object pose sequences from observations and use them to aid in action prediction, thereby enhancing the robustness and kinematic consistency of the policy’s observation-to-action mapping.
- We introduce MBA, a flexible auxiliary module that can be easily integrated into existing policies with diffusion heads, rather than functioning as an independent policy. Extensive experiments demonstrate its ability to significantly improve the performance of these policies, highlighting its broad applicability and potential to enhance a wide range of robotic manipulation tasks.

## II. RELATED WORK

### A. Imitation Learning for Robotic Manipulation

Imitation learning seeks to enable robots to acquire complex skills by learning from expert demonstrations. Recently, behavior cloning [29] from demonstrations has demonstrated promising performance across various robotic manipulation tasks [7, 10, 19, 22, 28, 61]. Most existing robotic manipulation policies are typically structured into three main components: an observation perception module for manipulation-relevant information extraction from observations, an optional policy backbone for processing the perception results, and an action head for generating actions based on the (processed) perception outputs.

For action generation, prior work has investigated both single- and multi-step action prediction methods. Single-step action prediction [7, 10, 22, 55, 61] is straightforward but often leads to inconsistencies between the consecutive actions. To address this limitation, several studies have applied the action chunking technique [56] to enable multi-step action sequence prediction. This includes the utilization of unimodal action heads, such as L1 and L2 action heads [4, 21, 24, 56], and multimodal action heads like the Gaussian mixture head [11, 23, 25, 33] and diffusion head [8, 28, 42, 46, 53]. Among these, the diffusion head [17] excels in capturing the diversity and complexity of action sequences, which is essential for executing real-world tasks.

Similar to the multimodal nature of robot action sequences in manipulation tasks, object motion sequences also exhibit diverse and complex characteristics, making diffusion heads well-suited for modeling such dynamics. Accordingly, we develop the MBA module, designed for seamless integration into policies with diffusion action heads [8, 42, 53]. This module enhances performance by decoupling the process into two stages: generating object motion via a motion diffusion head and conditionally generating robot actions through an action diffusion head.

### B. Object-Centric Manipulation Learning

Object-centric manipulation learning can be categorized into two main lines of approaches: one emphasizing scene understanding through structured, object-centric representations [12, 26, 39, 58–60], and the other focusing on action-oriented learning by identifying how objects can be manipulated to achieve specific goals [1, 3, 35, 49, 51].

The action-oriented approach actively engages with objects to achieve desired outcomes. Affordance learning is a central method here, aiming to identify possible actions based on an object’s physical properties and context [1, 5, 27, 35, 43, 52]. This approach, though effective in structured scenarios, often relies on predefined affordances, limiting its adaptability to dynamic environments and tasks that require fine-grained coordination of actions or adaptive responses.

To overcome these limitations, inspired by the success of flow-based methods [15, 37], recent work has utilized object flow as a general affordance for guiding manipulation policies [3, 13, 32, 40, 45, 49, 51, 54]. These policies map vision to motion by anchoring visual features and constructing object flow, which is then translated into robot actions using either heuristic transformations [3, 13, 32, 51, 54] or learned models [45, 49]. Many methods incorporate object pose as an intermediate representation [3, 32, 51] in such flow-to-action translations. However, using flow as an indirect motion representation can introduce ambiguity due to the vision-motion gap [3, 45, 49], complicating flow-to-action learning. Moreover, defining such heuristic transformations is challenging, often hindering generalization in dynamic or novel environments.

Our proposed MBA module employs object pose as a direct motion representation, which reduces training demands and enhances interpretability by establishing a clear link between visual inputs and motion outcomes. While a contemporaneous work [18] similarly leverages object pose for motion representation, it infers robot actions heuristically from object trajectories. We argue that generating actions from object motion guidance is essential, particularly in tasks involving deformable objects or dynamic conditions where heuristic methods may fall short.

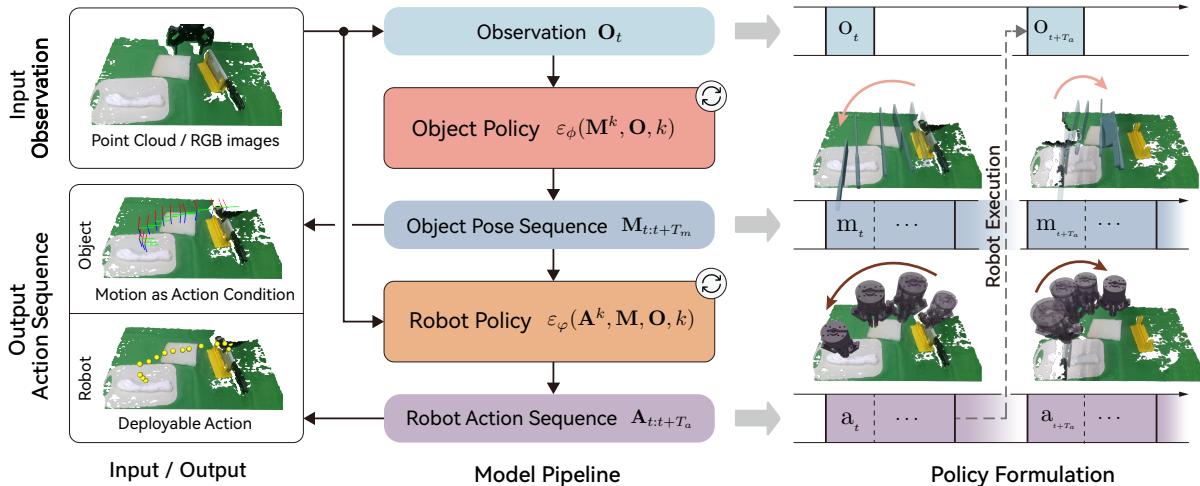
## III. METHOD

We aim to endow the policy with the capability to concurrently reason about object pose  $\mathbf{M}$  and robot action  $\mathbf{A}$  from observations  $\mathbf{O}$ . Specifically, we model the joint conditional distribution  $p(\mathbf{M}, \mathbf{A}|\mathbf{O})$  in MBA. This distribution can be decoupled as:

$$p(\mathbf{M}, \mathbf{A}|\mathbf{O}) = p(\mathbf{M}|\mathbf{O})p(\mathbf{A}|\mathbf{M}, \mathbf{O})$$

where  $p(\mathbf{M}|\mathbf{O})$  serves to sample the most likely object motions based on current observations, and  $p(\mathbf{A}|\mathbf{M}, \mathbf{O})$  serves to sample the most likely actions, guided by the sampled motions and current observations.

MBA has an object motion generation module  $p(\mathbf{M}|\mathbf{O})$  that is readily compatible with any existing policy with diffusion action heads, enhancing versatility and efficiency across various robotic manipulation contexts. Given an observation, the perception module of the original policy first processes it into observation features. These observation features are then



**Fig. 2: Overview of MBA pipeline.** MBA takes the current observation as input, which could be in the form of 3D point clouds or RGB images from different viewpoints. Object pose sequences are sampled as actions with denoising diffusion from the object policy to be part of the framework’s output. Conditioning on the observations and object pose actions, MBA samples deployable robot actions with denoising diffusion from the robot policy. These actions are executed within the workspace to update the environment state and the observations.

passed to MBA, where they serve as a condition for the object pose sequence denoising process. Subsequently, the diffusion action head of the original policy takes both the observation features and the denoised object pose sequence as inputs and outputs the corresponding robot actions. During training, both ground-truth object pose sequences and action sequences are used as supervision for the predicted object pose sequences from MBA and action sequences from the action head. At inference time, the policy with MBA operates in an end-to-end manner, without requiring ground-truth object poses. The full execution framework of MBA is illustrated in Fig. 2.

#### A. Object Motion Generation

As discussed in §II-B, we opt for object pose as a direct representation to describe object motion. The object pose  $m_t$  is represented as a 9D vector, combining 3D translation and 6D rotation [57], to align with robot action representations [8, 42, 53]. At each observation step  $t$ , we use the observation features  $O_t$  as the condition to the diffusion process [17], which denoises the object pose sequence for the next  $T_m$  steps  $M_{t:t+T_m} = (m_t, m_{t+1}, m_{t+2} \dots m_{t+T_m-1})$ .

Specifically, the diffusion-based object pose generation model begins by sampling an initial noise from a Gaussian distribution  $M^k \sim \mathcal{N}(0, I)$ . At each diffusion step  $k$ , the denoising network  $\varepsilon_\phi$  progressively removes this noise, conditioned on the observation features  $O_t$ . This process iterates to recover a noise-free, clean object pose sequence  $M^0$ :

$$M^{k-1} = \alpha_k (M^k - \gamma_k \varepsilon_\phi(M^k, O, k)) + \sigma_k \mathcal{N}(0, I), \quad (1)$$

where  $\{\alpha_k, \gamma_k, \sigma_k\}$  is the noise schedule.

To train the noise prediction network  $\varepsilon_\phi$ , we randomly pick a sequence of real object trajectory  $M^0$  and apply noise at a random diffusion step  $k$  through a forward diffusion process. The model is then tasked to predict the added noise  $\epsilon^k$  at the corresponding diffusion step. We employ mean squared error (MSE) loss as the objective function to supervise the prediction of object pose:

$$\mathcal{L} = \text{MSE}(\epsilon^k, \varepsilon_\phi(M^0 + \epsilon^k, O, k)), \quad (2)$$

where the ground-truth object pose  $M^0$  is obtained via a mo-

tion capture (MoCap) system during demonstration collection, and  $\epsilon^k$  is the added noises for the diffusion step  $k$ .

#### B. Robot Action Generation under Object Motion Guidance

Following [8, 42, 53], the robot action  $a_t$  is defined by the end-effector pose and gripper width, using a 10D representation that concatenates 3D translation, 6D rotation [57], and 1D gripper width together. Similarly, we use the diffusion process to generate the action sequence at the next  $T_a$  steps:  $A_{t:t+T_a} = (a_t, a_{t+1}, a_{t+2} \dots a_{t+T_a-1})$ .

In this process, we generate actions  $A^0$  using the same diffusion method as for object poses. Notably, the action noise prediction network receives not only the same observation features  $O$  as before, but also the object pose features  $M$  obtained in the previous stage. These features are concatenated to predict the noise jointly:

$$A^{k-1} = \alpha_k (A^k - \gamma_k \varepsilon_\phi(A^k, M, O, k)) + \sigma_k \mathcal{N}(0, I) \quad (3)$$

Given the physical and mathematical similarities between object poses and the robot’s end-effector poses, we use a similar encoding network for their feature representation. The object pose is unfolded along the  $T_m$  axis into  $T_m \times 9$  and then passed through MLP layers with dimensions  $(T_m \times 9, 32, 32)$ , ultimately resulting in the object pose feature vector  $M$ .

From every iteration, we collect an action sequence  $A^0$  from expert demonstrations and add noise over  $k$  steps. The noise prediction network is then tasked with predicting the noise based on the feature information, with supervision provided through MSE loss:

$$\mathcal{L} = \text{MSE}(\epsilon^k, \varepsilon_\varphi(A^0 + \epsilon^k, M, O, k)) \quad (4)$$

#### C. Execution

During execution, the policy with MBA estimates the object pose for  $T_m$  steps based on the observation inputs, which include environmental information and the robot state. This estimated pose sequence serves as a direct condition for generating an action sequence of  $T_a$  steps, without reliance on the MoCap system. The robot then executes up to  $T_a$  action steps before the next observation and execution cycle, with the exact number of steps depending on task type and

Method	<i>Adroit</i> (3)	<i>DexArt</i> (4)	<i>MetaWorld</i> Easy (28)	<i>MetaWorld</i> Medium (11)	<i>MetaWorld</i> Hard (6)	<i>MetaWorld</i> VeryHard (5)	Average (57)
DP3 [53]	$68.3 \pm 3.3$	$53.5 \pm 2.5$	$90.9 \pm 1.4$	$61.6 \pm 6.5$	$29.7 \pm 2.8$	$49.0 \pm 6.8$	$71.3 \pm 3.2$
DP3 w. MBA	$79.7 \pm 0.7$	$52.3 \pm 2.8$	$92.5 \pm 1.1$	$66.4 \pm 6.1$	$36.2 \pm 1.2$	$86.8 \pm 1.6$	$77.5 \pm 2.2$
DP [8]	$31.7 \pm 3.0$	$22.8 \pm 3.8$	$83.6 \pm 3.6$	$31.1 \pm 5.3$	$9.0 \pm 1.0$	$26.6 \pm 3.2$	$53.6 \pm 3.6$
DP w. MBA	$64.0 \pm 3.0$	$29.0 \pm 2.5$	$84.9 \pm 1.2$	$57.8 \pm 4.4$	$23.8 \pm 1.3$	$79.8 \pm 1.2$	$67.8 \pm 2.0$

**TABLE I: Performance comparison of policy models with and without MBA integration across 57 simulation tasks.** Average success rates ( $\pm$  standard deviation) are reported under three random seeds.

### Algorithm 1 MBA Execution Process

```

Require: Observation  $\mathbf{O}_t$ , Predetermined steps  $T_m, T'_a \leq T_a$ , Encoders  $\mathbf{F}_O, \mathbf{F}_M$ 
Ensure: Task completion via iterative action execution
while task is not complete do
    Encode the observation:  $O_t \leftarrow \mathbf{F}_O(\mathbf{O}_t)$ 
    Predict object pose sequence  $\{\mathbf{m}_t, \mathbf{m}_{t+1}, \dots, \mathbf{m}_{t+T_m}\}$ :
    for  $k = K$  downto 1 do
         $\mathbf{M}^{k-1} = \alpha_k(O^k - \gamma_k \epsilon_\phi(\mathbf{M}^k, O_t, k)) + \sigma_k \mathcal{N}(0, I)$ 
         $M_t \leftarrow \mathbf{F}_M(\mathbf{M}^0 = \{\mathbf{m}_t, \mathbf{m}_{t+1}, \dots, \mathbf{m}_{t+T_m}\})$ 
        Predict action sequence  $\{\mathbf{a}_t, \mathbf{a}_{t+1}, \dots, \mathbf{a}_{t+T_a}\}$  with extracted feature  $M_t$  and  $O_t$ :
        for  $k = K$  downto 1 do
             $\mathbf{A}^{k-1} = \alpha_k(A^k - \gamma_k \epsilon_\varphi(\mathbf{A}^k, M_t, O_t, k)) + \sigma_k \mathcal{N}(0, I)$ 
            Execute  $T'_a$  steps,  $T'_a \leq T_a$ 
            for  $i = 1$  to  $T'_a$  do
                Execute  $\mathbf{a}_{t+i}$ :  $\mathbf{a}_{t+i} \leftarrow \begin{cases} \mathbf{a}_{t+i} & \text{If task condition holds} \\ \text{Stop} & \text{Otherwise} \end{cases}$ 
            Update observations  $\mathbf{O}_t$ 
             $t \leftarrow t + T'_a$ 

```

environmental conditions. To adhere to standard kinematic principles, we ensure that  $T_m \geq T_a$  in all experiments. The specific process is detailed in Alg. 1.

## IV. SIMULATION EXPERIMENTS

In the simulation experiments, we aim to address the following research questions: **(Q1)** Can MBA effectively improve the performance of robotic manipulation policies by leveraging predicted object motion as a condition for generating robot actions? **(Q2)** Can the human-like reasoning process of MBA make policy learning more efficient?

### A. Setup

**Benchmarks.** We evaluate our policy in three simulation benchmarks, encompassing a total of 57 tasks:

- ***Adroit*** [30] utilizes a multi-fingered Shadow robot in the MuJoCo [38] environment to perform highly dexterous manipulation across a variety of tasks. These operations involve both articulated objects and rigid bodies.
- ***DexArt*** [2] uses the Allegro robot in the SAPIEN [47] environment to perform high-precision dexterous manipulation. It primarily focuses on tasks involving articulated object manipulation.
- ***MetaWorld*** [50] primarily operates in the MuJoCo environment, using a gripper to perform manipulation tasks involving both articulated and rigid objects. It covers a wide range of skills required for everyday scenarios, categorizing these tasks into difficulty levels: easy, medium, hard, and very hard.

**Baselines.** The focus of this work is to demonstrate that the introduction of MBA as a module can universally enhance the performance of existing policies with diffusion heads. Therefore, in our simulation experiments, we select representative Diffusion Policy (DP) [8], 3D Diffusion Policy (DP3) [53] as our 2D and 3D baselines. We then integrate the MBA module into these baselines and compare the performance. To ensure fairness in the experiments, we ensure that MBA and the corresponding baseline methods use the same expert demonstrations for training, with an equal amount of training steps. During the execution phase, both methods undergo the same number of observation and inference steps.

**Demonstrations.** In terms of expert demonstration generation, we employ scripted policies for ***MetaWorld***, the VRL3 [41] agent for ***Adroit***, and the PPO [31] agent for ***DexArt***. The average demonstration success rates for these agents are 98.7%, 72.8%, and nearly 100%, respectively. For ***Adroit*** and ***MetaWorld***, we use 10 demonstrations for training, while 100 demonstrations are used for ***DexArt***.

**Protocols.** Following [53], We conduct 3 runs for each experiment, using seed numbers 0, 1, and 2. For each seed, we evaluate 20 episodes every 200 training epochs as a test node, compute the average success rate over 20 episodes for each test node, and then average the top 5 performing test nodes. This accounts for the inherent instability of imitation learning policies during training in simulated environments, where different policies converge at varying training stages. We then report the mean and standard deviation of success rates across the 3 seeds.

### B. Results

**Integrating MBA results in more stable and superior performance (Q1).** We report the average success rates and standard deviations (across three random seeds) for all simulation tasks in the Adroit, Dexart, and MetaWorld environments, across four difficulty levels, in Table. I. On average, MBA outperforms the baseline in the majority of benchmarks, achieving a 14.2% increase in the average success rate over DP and a 6.2% increase over DP3. Additionally, we observe a general reduction in the average standard deviation of task execution across all benchmarks, further confirming the robustness of MBA. The results also demonstrate that MBA significantly improves success rates in tasks with higher difficulty levels. Detailed examples are provided in Table. II, where we report the average success rate and standard deviation for particular tasks. Notably, in tasks that require precise contact and fine manipulation within narrow action spaces, MBA can enhance

Method	Adroit				DexArt								MetaWorld					
	Door	Pen	Laptop	Toilet	Bin Picking	Box Close	Hammer	Peg Insert Side	Disassemble	Shelfplace	Reach							
DP3 [53]	62 ± 4	43 ± 6	81 ± 2	71 ± 3	34 ± 30	42 ± 3	76 ± 4	69 ± 7	69 ± 4	17 ± 10	24 ± 1							
DP3 w. MBA	74 ± 1	65 ± 1	78 ± 6	70 ± 2	54 ± 23	56 ± 2	98 ± 2	75 ± 5	98 ± 1	73 ± 1	32 ± 5							
DP [8]	37 ± 2	13 ± 2	31 ± 4	26 ± 8	15 ± 4	30 ± 5	15 ± 6	34 ± 7	43 ± 7	11 ± 3	18 ± 2							
DP w. MBA	45 ± 1	53 ± 3	42 ± 2	45 ± 2	45 ± 8	36 ± 7	89 ± 5	53 ± 0	71 ± 23	64 ± 2	26 ± 3							

TABLE II: Task-level success rates across 11 simulation tasks in 3 environments comparing MBA-augmented and baseline policies.

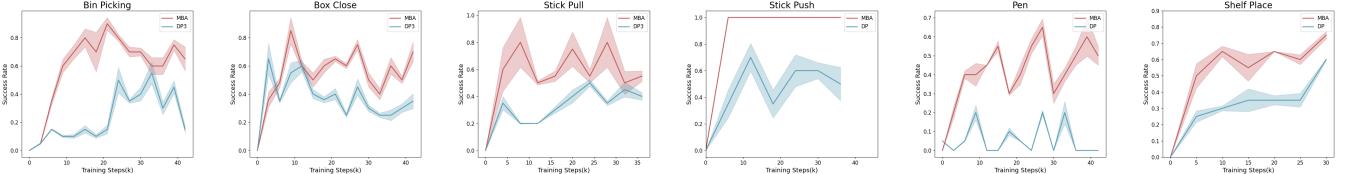


Fig. 3: Average learning curves (success rate - training steps) over three runs comparing MBA-augmented and baseline policies.

the execution capability and robustness of the robotic manipulation policies. We attribute this success primarily to two factors. First, when objects are stationary, MBA’s accurate prediction (can also be regarded as a pose estimation in this situation) of object pose enables the robot to localize and grip the objects effectively. Second, when objects are in motion, the strong correlation between the predicted object pose sequences and the robot’s action sequences provides robust guidance and calibration for action generation.

**MBA accelerates the policy learning process, leading to more efficient learning (Q2).** As shown in Fig. 3, we observe that policies with MBA exhibit higher learning efficiency during training compared to their vanilla counterparts. These policies typically reach the peak task test success rate at earlier training steps and maintain a more stable success rate at a higher level. This improvement is primarily attributed to the object pose information, which offers a more learnable and easily encoded feature representation.

## V. REAL-WORLD EXPERIMENTS

In the real-world experiments, we aim to evaluate the effectiveness of the proposed MBA module in enhancing the performance of real-world robotic manipulation policies across a variety of manipulation tasks.

### A. Setup

**Platform.** We employ a Flexiv Rizon robotic arm with a Robotiq 2F-85 gripper to manipulate objects. A global Intel RealSense D415 RGB-D camera is positioned in front of the robot for 3D perception, capturing single-view workspace point clouds. The robot workspace is defined as a 40 cm × 60 cm rectangular area in front of the robot. The overview of the real-world robot platform is shown in Fig. 4. All devices are connected to a workstation with an Intel i9-10980XE CPU and an NVIDIA 2080 Ti GPU for data collection and evaluation. Since the training process of MBA requires 6D object motion data for supervision, we set up a Motion Capture (MoCap) system comprising five OptiTrack Prime 13W infrared cameras. Notably, *no motion capture system or markers are required during deployment*.

**Tasks.** We select 4 tasks for experimentation: **Cut Clay** (a tool-use contact-rich task), **Put Bread into Pot** (soft body ma-

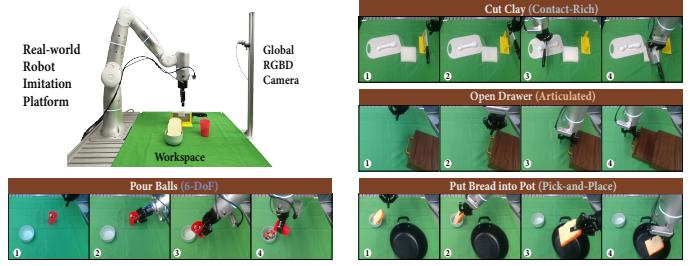


Fig. 4: Real-world deployment platform and execution process of four manipulation tasks.

nipulation), **Open Drawer** (articulated object manipulation), and **Pour Balls** (6-DoF task) as illustrated in Fig. 4.

**Demonstrations.** We collect 50 expert demonstrations through end-effector teleoperation with haptic devices for each task, following the same setup and procedures in [14, 42, 46].

**Baseline.** We retain DP and DP3 from the previous section as baselines, and additionally include RISE [42], a SOTA real-world policy model that operates solely on 3D scene input. RISE generates actions through a diffusion head, conditioned on 3D features extracted via sparse convolution networks [9] and transformers. We integrate the MBA module into these three policies, placing it before the action head to predict the future object pose sequence as the action condition.

**Protocols.** In the real-world evaluation, we conduct 20 trials per method for each task unless stated otherwise. All methods are compared under nearly identical randomized initial scene configurations for each trial.

### B. Cut Clay

The **Cut Clay** task consists of three stages: grasping the knife, slicing the clay until separation, and placing the knife on a foam mat. This task involves multiple stages, testing the ability of MBA to guide policy execution through accurate prediction of future object pose sequences.

In this experiment, we randomize the position and orientation of the cutting board, knife holder, and foam mat for each test. Special attention is given to varying the relative positions of these three objects to assess policy robustness thoroughly. Additionally, the shape of the clay is changed each time to meet the generalization requirements for real-world applications. We define five evaluation metrics for this task:

Method	Cut Clay				Put Bread into Pot		Open Drawer		Pour Balls		
	Pick (%)	Cut (%)	Sep. (%)	Place (%)	Attempt ↓	Succ. (%) ↑ Attempt ↓	Succ. (%) ↑	Pour (%)	Balls ↑	Pick (%)	
RISE [42]	95	70	30	50	11	80	7	37.5	85	8.15	85
RISE w. MBA	100	90	55	65	12	95	1	52.5	100	9.60	100
DP3 [53]	95	35	10	15	53	25	16	20	55	1.20	60
DP3 w. MBA	80	60	20	25	25	45	29	55	55	1.65	60
DP [8]	-	-	-	-	-	40	12	20	10	0.80	30
DP w. MBA	-	-	-	-	-	45	16	30	40	3.60	60

TABLE III: Performance comparison of policy models with and without MBA integration across four real-world tasks.

- Success rate of picking the knife.
- Success rate of completing the cutting motion.
- Success rate of slicing the clay until separation.
- Success rate of placing the knife on the mat.
- Total redundant knife-grasp attempts across 20 tests.

We observe limitations with 2D policies on this long-horizon multi-object task as the appearance of clay is similar in 2D visual observations before and after the cutting process. Therefore, we restrict our comparison to 3D policies. From Table. III, we observe that RISE with MBA shows significant improvements over RISE in all stages of task execution, particularly in the cutting and separation stage. A similar improvement is observed with DP3. This improvement is attributed to the need for rotating the cutting surface and splitting it when the knife cuts into the clay, which is inherently a 6-DoF task. MBA leverages the prediction of the tool motion to provide feedback on the execution of the current action, resulting in superior task performance compared to the baseline. However, MBA still exhibits repeated attempts to grasp the knife. We hypothesize that this issue arises from estimation errors in this task, where the thinness of the blade amplifies these errors.

#### C. Put Bread into Pot

The **Put Bread into Pot** task is a classic pick-and-place task, where the objective is to pick the bread from a bowl and place it into a pot. We choose this task because bread is a soft object. Unlike rigid bodies, soft objects can deform significantly due to the gripper's hold and the pressure from touching the bottom of the pot, which affects the object's pose, increasing the challenge for MBA in predicting the object pose sequence. We aim to assess the robustness and generalization ability of MBA through this task.

In this experiment, we randomly initialize the positions of the pot and the bowl and change the orientation of the bread within the bowl. We record two evaluation metrics: the average success rate of placing the bread into the pot and the total number of redundant attempts to grasp the bread.

The results, as shown in Table. III, indicate that RISE with MBA outperforms vanilla RISE by 15% in average success rate. In most cases, RISE with MBA successfully grasps the bread on the first attempt, whereas RISE often requires multiple attempts. This demonstrates that MBA can effectively handle object pose sequence prediction for soft objects, contributing to more accurate object localization and grasping.

Integrating MBA with both DP and DP3 also notably improves success rates. However, we also observe that a higher error rate in the base policy — primarily caused by the visual

encoder — leads to increased attempts by MBA. This is likely due to the error accumulates when the visual backbone has a significant error, necessitating more attempts to compensate.

#### D. Open Drawer

The **Open Drawer** task is a two-stage process where the robot must first precisely grasp the drawer handle and then pull it open horizontally. The main difficulty lies in the small distance between the handle and the drawer surface, where even slight positional errors can result in a failed grasp or cause the gripper to lose contact during the pulling motion. This task challenges the policy's ability to accurately estimate the handle pose sequence to ensure smooth and high-precision execution.

In this experiment, we account for the fact that delicate operations can be easily influenced by scene setup, particularly issues like point cloud occlusion when the robotic arm manipulates drawers at low heights. To minimize the impact of randomness, we double the number of test trials from the original plan, increasing it to 40 trials. Our test positions cover a variety of positions and orientations in the workspace, including areas within both the inner circle and the outer ring of the workspace plane. Each area receives 20 test trials. We also include scenarios where the initial drawer position causes point cloud occlusion, ensuring a comprehensive evaluation of the policies' robustness. We report the average success rates over all the trials.

The experimental results are shown in Table. III indicate that policies with MBA outperform the baselines in both stages of the task. This aligns with our findings in simulation experiments, further confirming that integrating MBA significantly enhances the policies in tasks that require precise object localization and fine manipulation.

#### E. Pour Balls

The **Pour Balls** task is a challenging one, where the objective is to lift a cup filled with 10 balls and pour them into a bowl. The difficulty arises from two main factors:

- 1) The cup is a non-cubic object, with varying width at different heights. Therefore, the gripper must learn a precise visual-to-motion control policy to grasp the cup at the correct height. If the gripper is too wide, it will fail to hold the cup, causing it to fall; if the gripper is too narrow, it will knock over the cup.
- 2) This is a 6-DoF task, and during the pouring process, if the translation and rotation are not properly controlled, or the gripper's force is not correctly adjusted, the cup will either rotate or fall, resulting in task failure.

Task	Metric	DP w. ATM [45]	DP w. MBA
<i>Put Bread into Pot</i>	Success (%) $\uparrow$	40	45
	Attempt $\downarrow$	2	16
<i>Open Drawer</i>	Success (%) $\uparrow$	5	30
	Pour (%)	25	40
<i>Pour Balls</i>	Balls $\uparrow$	2.05	3.60
	Pick (%)	35	60

TABLE IV: Comparison of policy models under different object motion conditions: MBA (ours): 6D poses vs. ATM: 2D point flow.

In this round of experiments, we still randomize the object positions. We conduct 20 trials. In this experiment, we consider three evaluation metrics: the average success rate of pouring balls into the bowl, the average number of balls poured into the bowl across all trials, and the average success rate of picking up the cup.

In this experiment, MBA achieves more than 15% the success rate compared to RISE and 30% compared to DP, as shown in Table III. Notably, under the same successful conditions of picking up the cup, MBA is also able to pour the balls with greater precision. This indicates that MBA can accurately capture the relationship between object pose variations and corresponding actions in 6-DoF tasks, making it well-suited for handling intricate manipulation scenarios.

#### F. Comparative Experiment to Flow-based methods

In this section, we compare MBA against the representative flow-based method ATM [45] to demonstrate that the pose prediction paradigm yields higher quality action generation for the policy compared to prediction in the visual space. Specifically, both ATM and MBA employ a diffusion action head for prediction; MBA conditions on predicted future object pose sequences, whereas ATM conditions on keypoint flow predicted by a Track Transformer.

We evaluate MBA on three 2D policy-fit tasks and found that it outperforms ATM across most metrics, as shown in Table IV. Notably, in tasks requiring fine-grained manipulation, such as *Open Drawer*, the flow-based approach’s tracking is confined to a limited number of pixels in the visual space, making it challenging to capture the pose of the handle, which can lead to grasping failures. In contrast, tasks with higher degrees of freedom, like *Pour Balls*, necessitate more comprehensive spatial motion modeling and precise control over the object’s rotational dynamics. In these scenarios, flow-based policies appear to be at a disadvantage. The results validate our hypothesis regarding the “vision-motion” gap, underscoring the necessity of modeling pose information.

#### G. Inference Speed

We evaluated the average inference time (excluding visual backbone processing time for experimental consistency across different backbones) for DP, ATM, and DP with MBA. DP achieved 95.98 ms, ATM 105.85 ms, while DP with MBA required 197.50 ms. This demonstrates the increased computational cost associated with MBA’s precise control. ATM benefits from its transformer-based flow tracking, avoiding computationally expensive denoising steps.

## VI. LIMITATIONS

MBA still exhibits several limitations. First, its inference efficiency can be improved. Future work might focus on replacing the existing denoising paradigm with more computationally efficient models [36, 56]. Second, MBA’s object pose supervision relies on ground-truth data from a MoCap system, which incurs high acquisition costs. Borrowing from flow-based methods that utilize flow generative models to provide ground truth [45, 51] offers a promising alternative. Correspondingly, leveraging foundation 6D pose estimation models [16, 44] for data annotation is also a valuable direction. Third, MBA is limited in handling varying numbers of objects, as its predefined observation vector constrains it to a fixed object count. Finally, the manipulation of deformable objects, whose 6D pose is not trackable, remains unaddressed. These limitations point to important directions for future research.

## VII. CONCLUSION

In this paper, we introduce MBA, a novel module that draws inspiration from the reasoning process of human beings. MBA infers future object motion sequences and uses them as guidance for action generation. It can be flexibly integrated into existing robotic manipulation policies with diffusion action heads in a plug-and-play manner, significantly improving the performance of these policies across a wide range of tasks. This work holds great potential for future development, including integrating it into other robotic manipulation policies with different action heads, utilizing diverse object motion demonstration data (e.g., human demonstrations or web videos) for supervision and learning, exploring its performance in long-horizon, multi-stage tasks for both object pose and action prediction, and expanding it into a general large-scale policy across multi-dataset, multi-task settings.

## REFERENCES

- [1] S. Bahl, R. Mendonca, L. Chen, U. Jain, and D. Pathak, “Affordances from human videos as a versatile representation for robotics,” in *CVPR*, 2023.
- [2] C. Bao, H. Xu, Y. Qin, and X. Wang, “Dexart: Benchmarking generalizable dexterous manipulation with articulated objects,” in *CVPR*, 2023.
- [3] H. Bharadhwaj, R. Mottaghi, A. Gupta, and S. Tulsiani, “Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation,” in *ECCV*, 2024.
- [4] H. Bharadhwaj, J. Vakil, M. Sharma, A. Gupta, S. Tulsiani, and V. Kumar, “Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking,” in *ICRA*, 2024.
- [5] J. Borja-Diaz, O. Mees, G. Kalweit, L. Hermann, J. Boedecker, and W. Burgard, “Affordance learning from play for sample-efficient policy learning,” in *ICRA*, 2022.
- [6] R. T. Born and D. C. Bradley, “Structure and function of visual area mt,” *Annu. Rev. Neurosci.*, 2005.
- [7] A. Brohan, N. Brown, J. Carbalajal, and ..., “RT-1: robotics transformer for real-world control at scale,” in *RSS*, 2023.
- [8] C. Chi and et al., “Diffusion policy: Visuomotor policy learning via action diffusion,” in *RSS*, 2023.
- [9] C. Choy, J. Gwak, and S. Savarese, “4d spatio-temporal convnets: Minkowski convolutional neural networks,” in *CVPR*, 2019.
- [10] O. X.-E. Collaboration *et al.*, “Open x-embodiment: Robotic learning datasets and rt-x models,” in *ICRA*, 2024.
- [11] Z. J. Cui, Y. Wang, N. M. M. Shafullah, and L. Pinto, “From play to policy: Conditional behavior generation from uncurated robot data,” in *ICLR*, 2023.
- [12] C. Devin, P. Abbeel, T. Darrell, and S. Levine, “Deep object-centric representations for generalizable robot learning,” in *ICRA*, 2018.

- [13] B. Eisner, H. Zhang, and D. Held, “Flowbot3d: Learning 3d articulation flow to manipulate articulated objects,” in *RSS*, 2022.
- [14] H.-S. Fang *et al.*, “Rh20t: A comprehensive robotic dataset for learning diverse skills in one-shot,” in *ICRA*, 2024.
- [15] A. Goyal *et al.*, “Ifor: Iterative flow minimization for robotic object rearrangement,” in *CVPR*, 2022.
- [16] O. Hirschorn and S. Avidan, *A graph-based approach for category-agnostic pose estimation*, 2024.
- [17] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” in *NeurIPS*, 2020.
- [18] C.-C. Hsu *et al.*, “Spot: Se(3) pose trajectory diffusion for object-centric manipulation,” *arXiv preprint arXiv:2411.00965*, 2024.
- [19] E. Jang and et al., “Bc-z: Zero-shot task generalization with robotic imitation learning,” in *CoRL*, 2021.
- [20] M. Janner, S. Levine, W. T. Freeman, J. B. Tenenbaum, C. Finn, and J. Wu, “Reasoning about physical interactions with object-oriented prediction and planning,” in *ICLR*, 2019.
- [21] S. Kareem *et al.*, “Egomimic: Scaling imitation learning via egocentric video,” *arXiv preprint arXiv:2410.24221*, 2024.
- [22] M. J. Kim *et al.*, “Openvla: An open-source vision-language-action model,” *arXiv preprint arXiv:2406.09246*, 2024.
- [23] S. Lee, Y. Wang, H. Etukuru, H. J. Kim, N. M. M. Shafullah, and L. Pinto, “Behavior generation with latent actions,” in *ICML*, 2024.
- [24] L. Liu *et al.*, “Foam: Foresight-augmented multi-task imitation policy for robotic manipulation,” *arXiv preprint arXiv:2409.19528*, 2024.
- [25] A. Mandlekar and et al., “What matters in learning from offline human demonstrations for robot manipulation,” in *CoRL*, 2021.
- [26] T. Migimatsu and J. Bohg, “Object-centric task and motion planning in dynamic environments,” *IEEE Robotics and Automation Letters*, 2020.
- [27] S. Nasiriany *et al.*, “Rt-affordance: Affordances are versatile intermediate representations for robot manipulation,” *arXiv preprint arXiv:2411.02704*, 2024.
- [28] Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, and ..., “Octo: An open-source generalist robot policy,” in *RSS*, 2024.
- [29] D. Pomerleau, “ALVINN: an autonomous land vehicle in a neural network,” in *NeurIPS*, 1988.
- [30] A. Rajeswaran *et al.*, “Learning complex dexterous manipulation with deep reinforcement learning and demonstrations,” *arXiv preprint arXiv:1709.10087*, 2017.
- [31] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv preprint arXiv:1707.06347*, 2017.
- [32] D. Seita, Y. Wang, S. J. Shetty, E. Y. Li, Z. Erickson, and D. Held, “Toolflownet: Robotic manipulation with tools via predicting tool flow from point clouds,” in *CoRL*, 2023.
- [33] N. M. Shafullah, Z. Cui, A. A. Altanzaya, and L. Pinto, “Behavior transformers: Cloning  $k$  modes with one stone,” in *NeurIPS*, 2022.
- [34] J. Song, C. Meng, and S. Ermon, “Denoising diffusion implicit models,” in *ICLR*, 2021.
- [35] M. K. Srirama, S. Dasari, S. Bahl, and A. Gupta, “Hrp: Human affordances for robotic pre-training,” in *RSS*, 2024.
- [36] Y. Su *et al.*, “Dense policy: Bidirectional autoregressive learning of actions,” *arXiv preprint arXiv:2503.13217*, 2025.
- [37] Z. Teed and J. Deng, “Raft: Recurrent all-pairs field transforms for optical flow,” in *ECCV*, 2020.
- [38] E. Todorov, T. Erez, and Y. Tassa, “Mujoco: A physics engine for model-based control,” in *IROS*, 2012.
- [39] S. Tyree *et al.*, “6-dof pose estimation of household objects for robotic manipulation: An accessible dataset and benchmark,” in *IROS*, 2022.
- [40] M. Vecerik *et al.*, “Robotap: Tracking arbitrary points for few-shot visual imitation,” in *ICRA*, 2024.
- [41] C. Wang, X. Luo, K. Ross, and D. Li, “Vrl3: A data-driven framework for visual deep reinforcement learning,” *NeurIPS*, 2022.
- [42] C. Wang, H. Fang, H.-S. Fang, and C. Lu, “Rise: 3d perception makes real-world robot imitation simple and effective,” in *IROS*, 2024.
- [43] X. Wang and et al., “Articulated object manipulation using online axis estimation with sam2-based tracking,” *arXiv preprint arXiv:2409.16287*, 2024.
- [44] B. Wen, W. Yang, J. Kautz, and S. Birchfield, “Foundationpose: Unified 6d pose estimation and tracking of novel objects,” in *CVPR*, 2024.
- [45] C. Wen *et al.*, “Any-point trajectory modeling for policy learning,” *arXiv preprint arXiv:2401.00025*, 2023.
- [46] S. Xia, H. Fang, H.-S. Fang, and C. Lu, “Cage: Causal attention enables data-efficient generalizable robotic manipulation,” *arXiv preprint arXiv:2410.14974*, 2024.
- [47] F. Xiang *et al.*, “Sapien: A simulated part-based interactive environment,” in *CVPR*, 2020.
- [48] J. Xu *et al.*, “An end-to-end differentiable framework for contact-aware robot design,” in *RSS*, 2021.
- [49] M. Xu *et al.*, “Flow as the cross-domain manipulation interface,” in *CoRL*, 2024.
- [50] T. Yu *et al.*, “Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning,” in *CoRL*, 2020.
- [51] C. Yuan, C. Wen, T. Zhang, and Y. Gao, “General flow as foundation affordance for scalable robot learning,” in *CoRL*, 2024.
- [52] W. Yuan *et al.*, “Robopoint: A vision-language model for spatial affordance prediction for robotics,” in *CoRL*, 2024.
- [53] Y. Ze and et al., “3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations,” in *RSS*, 2024.
- [54] H. Zhang, B. Eisner, and D. Held, “Flowbot++: Learning generalized articulated objects manipulation via articulation projection,” in *CoRL*, 2023.
- [55] T. Zhang, Y. Hu, J. You, and Y. Gao, “Leveraging locality to boost sample efficiency in robotic manipulation,” in *CoRL*, 2024.
- [56] T. Z. Zhao and et al., “Learning fine-grained bimanual manipulation with low-cost hardware,” in *RSS*, 2023.
- [57] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, “On the continuity of rotation representations in neural networks,” in *CVPR*, 2019.
- [58] Y. Zhu, Z. Jiang, P. Stone, and Y. Zhu, “Learning generalizable manipulation policies with object-centric 3d representations,” in *CoRL*, 2023.
- [59] Y. Zhu, A. Joshi, P. Stone, and Y. Zhu, “Viola: Object-centric imitation learning for vision-based robot manipulation,” in *CoRL*, 2022.
- [60] Y. Zhu, A. Lim, P. Stone, and Y. Zhu, “Vision-based manipulation from single human video with open-world object graphs,” *arXiv preprint arXiv:2405.20321*, 2024.
- [61] B. Zitkovich and T. Y. and..., “RT-2: vision-language-action models transfer web knowledge to robotic control,” in *CoRL*, 2023.