2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

MetaDP: Meta-manipulation Diffusion Policy for Robotic Manipulation

Zheyi Zhao[1,2], Ying He[1,†], F. Richard Yu[3], Jiyuan Song[2], and Xilong Xun[2]

***Abstract*— In the field of 3D object manipulation, collecting expert data for end-to-end imitation learning is a standard approach. While successful, previous works have not adequately determined the specific phase of the current task, leading to low tolerance for feature variability and difficulty in addressing compounding errors, which are issues particularly prominent in high-precision tasks. To address these limitations, we introduce a novel framework named Meta-manipulation Diffusion Policy (MetaDP). This framework utilizes meta-manipulation prompt vectors to inform the model of the current stage of the task, which, in conjunction with noise reduction levels, controls the denoising process of the diffusion policy, thereby enhancing the model's predictive accuracy. Comprehensive experiments demonstrate that MetaDP significantly surpasses established baselines, achieving a relative improvement of 5% across eight complex tasks, which underscores the superiority of our method.**

I. Introduction

In the field of 3D object manipulation, collecting expert data for end-to-end imitation learning is a standard approach. This strategy facilitates the direct replication of expert techniques, which is crucial for advancing manipulation models. Three main types of methods have been studied for this approach. The first aims to enhance the representation of 3D information, as seen in approaches such as RVT [1]. The second focuses on diffusion models, which fit the distribution between environments and actions across different tasks, as exemplified by DP [2]. The third approach, based on large models, includes examples such as OpenVLA [3], which leverages advanced reasoning and inference to predict the next action.

Although previous methods have achieved significant success, they have not adequately addressed the issue of compounding errors. Since imitation learning is a form of discrete learning and lacks data on minor errors, models often incorrectly assume that objects in the gripper are perfectly secured and correctly oriented. Compounding errors typically occur in testing environments, where each predicted step by the model deviates slightly from the expected position. These deviations accumulate, eventually hindering task progression. This problem is especially pronounced in high-precision tasks, which require low motion error tolerance and thus are more likely to diverge from the data distribution observed in the training set. This implies that models should prioritize current conditions and ideally minimize the influence of historical data. Additionally, a clear understanding of the current phase of the task is essential to determine the appropriate next action step.

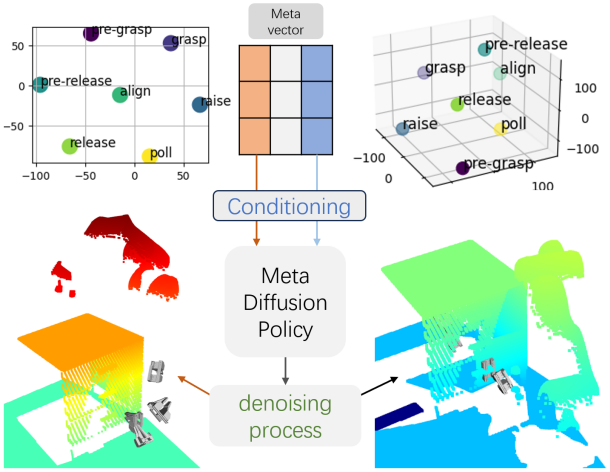

Fig. 1: **The overview of our key innovation for addressing compounding errors.** Our framework uses meta-manipulation vectors to indicate task stages. Projected in 2D/3D via t-SNE, they show a well-separated distribution, demonstrating that the model learns distinct phase characteristics, with different vectors activating features in each phase.

Previous methods for assessing the current phase of a task have typically relied on historical images or past actions as cues, which not only fall short in high-precision tasks but also exacerbate error accumulation and increase model complexity, thereby reducing generalizability. One approach involves using task timesteps to indicate the task's stage. However, this method faces challenges as the same task can be completed in varying numbers of steps, rendering timesteps a somewhat unreliable measure. Moreover, treating timesteps solely as individual tokens can lead to inadequate attention from the model. Furthermore, merging these timestep tokens with other feature tokens could distort their inherent information, complicating the training process.

To address these challenges, this paper introduces a novel Meta-manipulation Diffusion Policy (MetaDP) that innovates from the perspective of conditional variables. We propose that task completion by a robotic arm consists of multiple stages, each beginning with a keypose and ending with the next. The actions occurring between these keyposes are termed meta-manipulations, which can be labeled based on

† Corresponding author: Ying He, heying@szu.edu.cn.
This work is supported in part by Shenzhen Science and Technology Program under Grant ZDSYS20220527171400002, the National Natural Science Foundation of China (NSFC) under Grants 62271324, 62231020, 62394335 and 62371309, and the Open Research Fund from Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ) (Grant No. GML-LF-24-32)
[1]College of Computer Science and Software Engineering, Shenzhen University, China.
[2]Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen, China.
[3]School of Information Technology, Carleton University, Ottawa, Canada.

their physical significance with tags such as pre-grasp, grasp, pre-release, release, etc.

MetaDP utilizes meta-manipulation prompt vectors to inform the model of the current task stage, effectively signaling the model to "prepare to do something". These vectors, combined with noise reduction levels, control the denoising process in the diffusion policy, thereby improving predictive accuracy, addressing compounding errors, and enhancing the model's capability to execute high-precision tasks.

Comprehensive experiments demonstrate that MetaDP significantly surpasses established baselines, achieving a relative improvement of 5% across eight complex tasks and underscoring the superiority of our method.

To summarize, our contributions are threefold:

- We introduce a pioneering framework named ***MetaDP***, marking the inaugural integration of "meta-manipulation prompt vectors" within a model. This novel concept enhances the model's ability to recognize the current task stage.
- To mitigate compounding errors, the MetaDP framework leverages meta-manipulation vectors in synergy with noise reduction levels to control the denoising process of the diffusion policy, significantly enhancing the model's predictive accuracy.
- Extensive experiments show that our MetaDP can outperform the baselines, showing promising results in simulations.

II. Related Work

A. Robotic Manipulation in 3D

Recent work explores diverse perceptual representations for 3D manipulation. Image-based approaches are adopted in RT-1 [4] and ACT [5], while PolarNet [6] and M2T2 [7] process RGB-D point clouds with encoder–transformer architectures. PerAct [8] and FourTran [9] voxelize point clouds for 3D convolution, and Act3D [10], ChainedDiffuser [11], and RVT [1] exploit multi-scale or multi-view 3D representations. Beyond perception, some works enhance object grounding and reasoning. TagGuideBot [12] employs explicit visual prompts, whereas ABM [13] leverages feature-level representations for grounding. RT-2 [14], RT-H [15], and OpenVLA [3] integrate visual inputs into language models for action prediction, while VoxPoser [16] and CoPa [17] exploit code and tool-use capabilities. RT-H [15] and LLakey [18] further decompose tasks into subtasks and propagate intermediate instructions to downstream models. Inspired by this paradigm, we introduce an implicit conditional vector as a task-phase indicator to guide more precise outputs.

B. Diffusion Models for Robotics

UniversalPolicy [19] employs text-conditioned video generation and an inverse dynamics model to produce robotic arm state-action pairs. DiffusionRosie [20] applies diffusion-based data augmentation with semantic editing. SynthER [21] trains a standalone diffusion model to generate higher-quality samples than standard data augmentation. DBC [22] jointly optimizes imitation learning and diffusion losses. PlanningDiffusion [23] encodes state-action trajectories into 2D arrays for diffusion-based optimization. [24] forecasts future states and obtains actions via inverse dynamics. DiffusionPolicy [2] represents an alternative to imitation learning, where the condition is an image sequence, and it generates an action sequence. CrosswayDiffusion [25] employs a combination of reconstruction loss and diffusion loss. These diverse approaches leverage diffusion models to enhance policy training, data augmentation, and action sequence generation, contributing to more robust and efficient robotic manipulation. However, most methods rely on historical observations or action tokens as conditional vectors to guide the denoising process. These continuous variables are difficult to distinguish, which increases the training complexity and makes it challenging to address compounding errors and high-precision tasks. In contrast, we use discrete vectors to represent different task phases, ensuring greater separation between them, which facilitates model distinction. Additionally, we have innovatively integrated current gripper information, preventing the model from engaging in shortcut learning.

C. High Precision Manipulation

Previous work has leveraged various sensory inputs for high-precision robotic tasks. For instance, proprioception data supports peg-in-hole policies via imitation learning [26] and final-inch insertion via reinforcement learning [27]. Force-torque [28] and vision-based tactile sensors [29] further boost execution accuracy [30]. ACT [5] predicts entire action sequences rather than single actions, then ensembles overlapping chunks for smoother trajectories. RVT2 [31] uses virtual views around the robot to locate the area of interest before zooming in for precise gripper pose prediction. Similar to RVT2, we also predict the next keypose, but we incorporate discrete conditional vectors into the denoising process for more precise control and better handling of task phases.

III. Method

The architecture of the Meta-manipulation Diffusion Policy is shown in Figure 2. It is a conditional diffusion model that takes as input visual observations, a language goal, the current robot's end-effector's pose, and the current estimate for the robot's next keypose, and predicts the error in the end-effector's 3D translations and 3D orientations for each predicted timestep. We review the Denoising Diffusion Probabilistic models in Section 3.1 A and describe the architecture of our model in Section 3.2 B. For convenience, we will refer to the meta-manipulation prompt vector as the **meta vector** in the following sections.

A. Denoising Diffusion Probabilistic Models

A diffusion model approximates a probability distribution $p(x)$ by reversing a procedure that incrementally introduces noise to a sample x.

The diffusion process is governed by a variance schedule $\{\beta_t \in (0,1)\}_{t=1}^{T}$, which determines the amount of noise

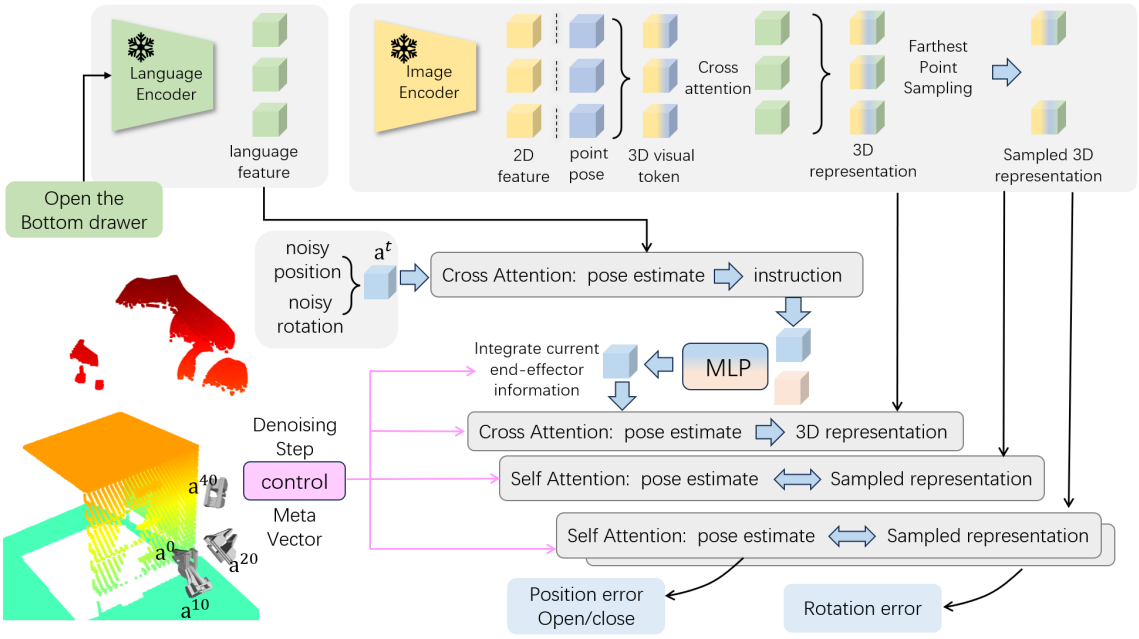

Fig. 2: **Overview of our proposed MetaDP.** We first obtain a multi-scale visual feature token, linking 2D visual tokens with point positions to form 3D tokens. These interact with the language feature via cross-attention for a comprehensive 3D representation. Noisy position and rotation data are projected into a higher-dimensional space with MLPs, interact with the language feature, and fuse with the gripper feature vector. This fused representation undergoes cross-attention with the full 3D representation and self-attention with the sampled one. Finally, the updated next keypose feature is used in MLPs to predict position error, rotation error, and gripper openness.

introduced at each time step. The noisy version of a sample x at time t can then be written as $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$ where $\epsilon \sim \mathcal{N}(0,1)$, is a sample from a Gaussian distribution (with the same dimensionality as x), $\alpha_t = 1 - \beta_t$, and $\tilde{\alpha}_t = \prod_{i=1}^{t} \alpha_i$.

The denoising process is represented by a neural network $\hat{\epsilon} = \epsilon_\theta(x_t; t)$, which receives the noisy sample x_t and the noise level t as inputs and aims to estimate the noise component ϵ.

Diffusion models can be readily adapted to generate samples from a conditional distribution $p(x|\mathbf{e})$ by incorporating the input $\mathbf{e}$ into the network ϵ_θ.

For us $\mathbf{e}$ is the observation, which is the visual scene captured by multiple calibrated RGB-D images o, and a language goal g. We change $\hat{\epsilon} = \epsilon_\theta(x_t; t)$ to $\hat{\epsilon} = \epsilon_\theta(x_t; m, t)$, m is the meta-manipulation prompt vector. Given a collection of $D = \{(x^i, \mathbf{e}^i)\}_{i=1}^{N}$ of end-effector keyposes trajectories x^i paired with observation $\mathbf{e}$ and meta-manipulation prompt vector m^i, along with the initial state $\{x^0, \mathbf{e}^0, m^0\}$ the noise prediction network is asked to predict the noise from the data sample with noise added.

$$L = MSE(\epsilon_\theta(\sqrt{\bar{\alpha}_t x^i} + \sqrt{1-\bar{\alpha}_t}\epsilon, x^{i-1}, \mathbf{e}^{i-1}, m^{i-1}, t), \epsilon) \tag{1}$$

As shown in [32], minimizing the loss function in Equation 1 also minimizes the variational lower bound of the KL-divergence between the data distribution and the distribution of samples drawn from the DDPM $p(x)$.

Starting from x_T sampled from Gaussian noise, the DDPM performs T iterations of denoising to produce a series of intermediate actions with decreasing levels of noise, $x_t, x_{t-1}, \ldots, x_0$, until a desired noise-free output x_0 is formed according to a specified sampling schedule, terminating with x_0 sampled from $p_\theta(x)$.

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, x^{i-1}, o^{i-1}, g, m^{i-1}, t)\right) + \frac{1-\bar{\alpha}_{t+1}}{1-\bar{\alpha}_t}\beta_t z, \tag{2}$$

where $z \sim \mathcal{N}(0,1)$.

B. Meta-manipulation Diffusion Policy

Dataset Construction. We have a dataset of demonstrations $\mathcal{D} = \{(\xi^1, g^1), (\xi^2, g^2), \ldots, (\xi^n, g^n)\}$, comprising n instances of expert demonstrations, each correlated with English language goals $\mathcal{G} = \{g^1, g^2, \ldots, g^n\}$. These demonstrations have been curated by an expert leveraging a motion planner to attain specific intermediate poses. Each demonstration ξ consists of a series of continuous actions $\mathcal{A} = \{a^1, a^2, \ldots, a^m\}$ alongside corresponding observations $\mathcal{O} = \{o^1, o^2, \ldots, o^i\}$. Observations o entail RGB-D imagery from an array of cameras, utilizing four cameras in simulated experiments $o_{\text{sim}} = \{o_{\text{front}}, o_{\text{left}}, o_{\text{right}}, o_{\text{wrist}}\}$

Each action a describes an end-effector pose and is decomposed into 3D position, 3D orientation and a binary open/closed state: $a = \{a^{\text{pos}} \in R^3, a^{\text{rot}} \in R^6, a^{\text{open}} \in \{0,1\}\}$. We represent rotations using the 6D rotation representation of [33].

In every demonstration ξ, we use the change in the open/close state of the gripper and the condition where the

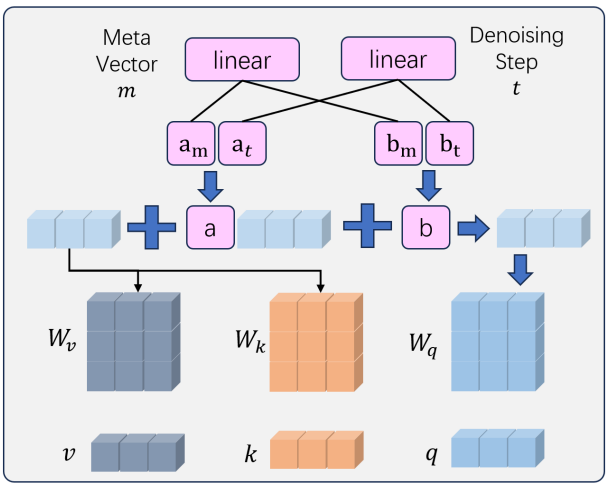

Fig. 3: **The overview of our control module.** The control module takes the meta vector m and the denoising step t as inputs, processing them separately through linear projectors to generate their respective scaling and offset values. These are then combined to produce the final scaling and offset parameters.

velocity remains zero for four consecutive frames as criteria to identify a subset of pivotal actions $\{k^1, k^2, \ldots, k^n\} \subset \mathcal{A}$ as keyposes. Keyposes are important intermediate end-effector poses that summarize a demonstration. Additionally, We designate the initial action of each demonstration as k^0, which serves as the zeroth keypose.

Each keypose action k is linked to a specific meta manipulation prompt such as pre-grasp, grasp, pre-release, etc., to indicate that k is the starting point of this stage. When input into the model, these prompts are mapped to their corresponding embedding vectors, where each embedding vector m^i is a trainable parameter, denoted as $\{m^1, m^2, \ldots, m^n\} \subset \mathcal{M}$. This approach allows us to form a training dataset consisting of tuples $(k^n, o^n, m^n, g, k^{n+1})$, where each tuple is derived from the sequence of demonstrations. The ultimate goal of the training is for the model to accept inputs (k^n, o^n, m^n, g) and predict the next keypose k^{n+1}.

Scene and language encoder. Our approach to processing RGB-D environmental information and language information is consistent with the method used by the 3D Diffuser Actor. Initially, we utilize a pre-trained 2D feature encoder and a feature pyramid network to obtain a multi-scale visual feature token. We apply the same intensity scaling to the point cloud, linking each 2D visual feature token with point positions to construct a 3D visual feature token and achieve a 3D representation. Noisy next keypose estimates are projected into high-dimensional vectors by MLP and are regarded as 3D visual feature tokens to interact with environmental information. Due to a large number of 3D visual feature tokens, the next keypose estimate can only perform cross-attention with them. Farthest Point Sampling is employed to select a subset of 3D representation for self-attention with the next keypose estimate to enhance information fusion.

Denoising Transformer. The MetaDP iteratively denoises an estimate of the end-effector's next keypose. Each keypose step token is represented as a 10-dimensional action, as previously defined.

Although the current pose of the end-effector is crucial, directly incorporating this information into the model during training could lead to the model taking shortcuts. Specifically, the model might learn to predict the next keypose by focusing solely on the difference between the current pose and the next keypose, rather than understanding the overall context of the scene. For example, during the "grasp" phase, the model might only learn to adjust one dimension (x, y, or z), or during the "rotate" phase, it might only focus on rotational changes. This narrow focus could cause the model to ignore important scene information, leading to poor generalization. As a result, the model may lose its ability to correct errors if the initial pose of the end-effector deviates from the ideal during testing, ultimately leading to task failure.

To address this issue, we introduce a learnable vector with dimensions consistent with the 3D visual features. This vector is also regarded as a 3D representation token. Using the current end-effector's position as this feature's location in the point cloud space, it performs cross-attention with the full 3D representation to integrate scene information. Finally, we fuse it with the next keypose estimate vector through an MLP, indirectly incorporating the current end-effector's information and preventing shortcut learning by the model. Up to this point, the current end-effector, visual information, and the next keypose estimates a_t^{pos}, a_t^{rot} are all projected into a high-dimensional space and treated as 3D visual feature tokens. Each is assigned a position embedding based on their respective positions in the point cloud. They interact within 3D relative position attention layers and utilize rotary positional embeddings as employed in the [34], with the property that the dot product of two positionally encoded features x_i, x_j is:

$$PE(p_i, x_i)^T PE(p_j, x_j) = x_i^T M(p_i - p_j) x_j \tag{3}$$

where $PE(p_i, x_i)$ and $PE(p_j, x_j)$ are the rotary positional embeddings of the features x_i and x_j concerning their positions p_i and p_j, and M is a matrix function that only depends on the relative positions of the points. The interaction with language g will use a standard attention mechanism.

All attention layers that compute attention with the 3D visual token are controlled by the denoising level timestep t and the meta vector m. We adopt the underlying principles of FiLM for dual control by scaling and shifting the features used to compute the query vector in the attention layers, thereby achieving the desired control effect, as illustrated in Figure 4. The specific control formula is expressed as follows:

$$Q_x = W_q(x + \gamma(t, m) \cdot x + \beta(t, m)) \tag{4}$$

where Q_x represents the query vector of the current token in the attention layer, and $\gamma(t, m)$ and $\beta(t, m)$ are modulation parameters adjusted based on the denoising timestep t and meta vector m.

After the next keypose estimate token interacts with various feature tokens through attention mechanisms, the updated next keypose feature token is fed into MLPs to predict the position error $\epsilon_\theta^{pos}(a_t^{pos}, a_t^{rot}, x^{i-1}, o^{i-1}, l, m^{i-1}, t)$, the rotation error $\epsilon_\theta^{rot}(a_t^{pos}, a_t^{rot}, x^{i-1}, o^{i-1}, l, m^{i-1}, t)$, and whether the end-effector should be open or closed $f_\theta^{open}(a_t^{pos}, a_t^{rot}, x^{i-1}, o^{i-1}, l, m^{i-1}, t)$

$$a_{t-1}^{pos} = \frac{1}{\sqrt{\alpha_t}}\Big(a_t^{pos} - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta^{pos}(a_t^{pos}, a_t^{rot}, x^{i-1}, o^{i-1}, g, m^{i-1}, t)\Big) + \frac{1-\overline{\alpha}_{t+1}}{1-\overline{\alpha}_t}\beta_t z^{pos}, \tag{5}$$

$$a_{t-1}^{rot} = \frac{1}{\sqrt{\alpha_t}}\Big(a_t^{rot} - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta^{rot}(a_t^{pos}, a_t^{rot}, x^{i-1}, o^{i-1}, g, m^{i-1}, t)\Big) + \frac{1-\overline{\alpha}_{t+1}}{1-\overline{\alpha}_t}\beta_t z^{rot}, \tag{6}$$

where $z^{pos}, z^{rot} \sim \mathcal{N}(0, 1)$ variables of appropriate dimension. We use the following two noise schedulers as in [34]:

1) a scaled-linear noise scheduler $\beta_t = (\beta_{max} - \beta_{min})t + \beta_{min}$, where β_{max}, β_{min} are hyperparameters, set to 0.02 and 0.0001.

2) a squared cosine noise scheduler $\beta_t = \frac{1-\cos\left(\frac{(t+1/T+0.008)}{1.008} * \frac{\pi}{2}\right)^2}{\cos\left(\frac{t/T+0.008}{1.008} * \frac{\pi}{2}\right)^2}$.

C. Intropy Analysis of MetaDP

Intropy measures learning efficiency as $\mathrm{d}L = \delta S/R$, where δS is effective performance improvement and R denotes learning resistance

Intropy is defined as a measure of intelligence efficiency [35], [36], quantified as the ratio between effective performance improvement and the resistance encountered during learning and inference, $\mathrm{d}L = \delta S/R$, where δS denotes the reduction in task-relevant discrepancy (e.g., prediction or control error) and R represents effective resistance arising from uncertainty, data inefficiency, computational cost, or structural mismatch.

From this perspective, MetaDP enhances learning efficiency by reducing resistance associated with task-phase ambiguity in diffusion-based imitation learning. The introduction of discrete meta-manipulation prompt vectors constrains the denoising process to phase-consistent action subspaces, effectively narrowing the hypothesis space explored during sampling. This structural bias mitigates compounding errors and improves robustness in high-precision manipulation tasks. Consequently, MetaDP achieves greater discrepancy reduction per unit of resource, indicating higher Intropy. Under the Intropy framework, its performance gains arise from principled resistance reduction rather than increased model scale.

IV. Experiments

Environment. The simulation is conducted in CoppelaSim [37] and interfaced via PyRep [38]. All experiments utilize a Franka Panda robot equipped with a parallel gripper. Input observations are obtained from four RGB-D cameras located at the front, left shoulder, right shoulder, and wrist. Using these cameras and their parameters, point clouds can be constructed in the world coordinate system. All cameras are noiseless and have a resolution of 256 x 256 pixels.

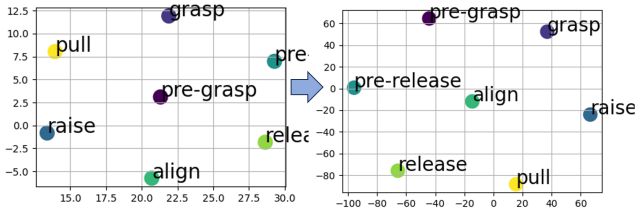

Fig. 4: **2D Visualization of Meta Vector.** The left image represents the 2D visualization of the meta vectors after dimensionality reduction using t-SNE at the network's initialization, while the right image shows the 2D visualization of the meta vectors, also reduced by t-SNE, after the training is completed.

Tasks. We train and evaluate eight RLBench [39] tasks from the publicly available Peract [8] dataset, with images re-rendered to 256×256 pixels, all challenging. These include four high-precision tasks: insert pegs, sort shapes, stack cups, and screw bulbs. In the sort shape task, the robot must pick up a specifically shaped object from the tabletop and place it into the shape sorter. This task effectively tests the precision of the learned model since the clearance between the object and the shape sorter is very tight, requiring perfect alignment; any minor error results in a failed insertion. In the stack cups task, a minor error in the pick-and-place locations of the cup results in failure, as evidenced by the low success rate of prior methods. Similarly, in the screw bulb task, the bulb's base must be well aligned with the socket for successful screwing. Additionally, there are four non-high-precision tasks: put in the cupboard (a scene understanding task), stack blocks (a compositional task), and two general tasks, close jar, and open drawer. Each task includes several variations, ranging from 3 to 60 possibilities. For example, in the sort shape task, "put the cube in the shape sorter" and "put the moon in the shape sorter" are two variants. Within the same variant, each episode only varies in the pose and rotation of the objects and target positions.

Keyposes and Meta Vector. In an expert trajectory, the keypose is defined as the pose of the robotic arm and gripper at frames where the joint velocity of the robotic arm is zero for four consecutive frames or where there is a change in the gripper's state. The action trajectory from k_n to k_{n+1} is referred to as meta-manipulation. Meta-manipulations with the same physical significance across different tasks share the same natural language prompt, such as pre-grasp. When input into the model, these prompts are mapped to their corresponding embedding vectors. Although each task has different variations, the sequence of meta-manipulations within the same task remains unchanged. Consequently, we only need to label each task manually once, resulting in low labor costs.

Training and Evaluation Details. For the eight tasks, each

TABLE I: **Multi-Task Performance on 8 RLbench tasks**. The first four columns represent High-Precision Tasks, while the last four columns represent General Tasks. Each agent is tested five times on these eight tasks, with the median result taken as the outcome.

Task Type	Model	insert Peg	screw bulb	sort shape	stack cup	close jar	open drawer	put in cupboard	stack blocks	Average
Multi-Task performance (three camera)	RVT	16	48	32	32	48	80	56	28	42.5
	RVT2	28	**88**	36	48	100	72	60	48	60
	3D Diffuser Actor	72	80	40	44	100	84	76	**54**	68.75
	Meta-manipulation Diffusion Policy (ours)	52	84	**60**	**64**	**100**	**100**	**80**	50	**73.75**
Multi-Task performance (one camera)	RVT	0	40	16	24	72	20	0	28	25
	RVT2	0	4	0	0	0	0	0	0	0.5
	3D Diffuser Actor	37.3	77.6	40	**64**	92	77.6	**72**	66	65.8
	Meta-manipulation Diffusion Policy (ours)	**41.3**	**80**	**44**	56	**96**	**100**	65	**70**	**69**

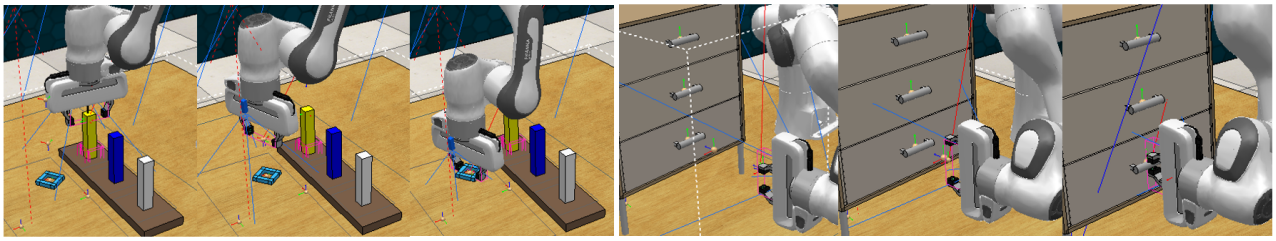

Fig. 5: **Generalization Capability of MetaDP.** In the task shown in the left image, the object fails to be placed and bounces out, creating a scenario where both the gripper state and environment are completely unfamiliar. Despite this, MetaDP, guided by the meta vector, can perform zero-shot decision-making. Similarly, in the task depicted in the right image, the gripper slips while pulling out a drawer, another scenario the model has never encountered before. Yet, MetaDP is able to re-execute the task. These examples demonstrate the strong generalization capability of MetaDP.

task generates one hundred episodes. After keyframe extraction, each episode yields 6 to 24 keyframes. We augment RGB-D observations with random rescaling and cropping, using nearest neighbor interpolation for rescaling. Training is conducted on four A800 GPUs, each with a batch size of 12, for a total of 480,000 steps. Each multi-task agent is evaluated independently on all eight tasks. Evaluations are scored as either 0 for failures or 100 for complete successes, with no partial credits. During evaluation, an agent continues to take actions until an oracle indicates task completion or until a maximum of 25 steps is reached. In the testing environment, meta-manipulation prompts are sequentially applied according to predefined rules. For example, in the "open drawer" task, prompts are applied in order as "pre-grasp", "grasp", "pull", "pre-grasp", and so on, until either 25 steps are completed or the simulation environment determines the task is finished. Each multi-task agent is tested five times on each task, and the median score is taken as the final result.

Baselines. We consider the following baselines:

1) RVT, a 3D policy that deploys a multi-view transformer to predict actions and fuses those across views by back-projecting to 3D.

2) RVT2, a 3D policy for high-precision tasks, first identifying the area of interest using virtual views and then refining the gripper pose with zoomed-in views.

3) 3D Diffuser Actor, a diffusion policy that lifts 2D features to 3D, predicting end-effector pose errors at each denoising step based on prior poses and denoising levels.

Given that we selected eight tasks, we adjusted the baseline training setup by changing only the training tasks while keeping the dataset and all other settings the same. The baseline was retrained under these conditions. During testing, each task was evaluated five times, and the median score was used as the final result for the model on that task.

Experiment Results. We show quantitative results in Table I. MetaDP achieves an average success rate of 65% across four high-precision tasks, surpassing the 3D Diffuser Actor, a diffusion policy that conditions on the history of end-effector poses, by 6%, and outperforming RVT2, a model specifically designed to address high-precision tasks, by 15%. Moreover, we observe substantial improvements in specific tasks, with success rate increases of 20% in sorting shapes and 16% in stacking cups, tasks that most baseline models fail to complete. Additionally, across four other tasks, MetaDP achieves an average success rate of 82.5%, exceeding state-of-the-art models by 4%. The task of opening a drawer consistently achieves a 100% success rate, attributable to the MetaDP's generalization capabilities demonstrated in Figure 5. Ultimately, the MetaDP achieved a success rate of 73.5%. Under the condition of a monocular camera, MetaDP

TABLE II: **Ablation study.** We maintain the same simulation setup as previously mentioned and report the average success rate on the eight RLBench tasks selected earlier.

Model	Avg. Success.
MetaDP w/o Meta Vector	66.75
MetaDP w/o Current Endpos	71.75
MetaDP with D. T.	73.75
MetaDP (ours)	73.75

achieves an average success rate of 55.3% across four high-precision tasks. For the other four tasks, the average success rate reaches 82.75%, resulting in an overall average success rate of 69%, which also surpasses the baselines. these results further validate the effectiveness of our approach, underscoring its robust performance and reliability in a wide range of tasks.

Ablations. We also compare to the following ablative versions of our model:

1) MetaDP w/o Meta Vector: This version of the policy eliminates the meta vector, relying solely on the denoising timestep to control the denoising process. By removing the meta vector, the model simplifies its input but focuses more directly on temporal progression during denoising.

2) MetaDP w/o Current Endpos: In this variant, the learnable variable associated with the gripper's position is removed. This modification prevents vector fusion with the current estimate position vector. However, the model continues to use the meta vector and denoising timestep to manage the denoising process, ensuring that other contextual factors still guide the noise reduction.

3) MetaDP with D. T.: This policy modifies the training approach. It first trains on six tasks until convergence, then introduces additional tasks one by one, each trained to convergence. The purpose of this staggered training approach is to observe whether the meta vectors maintain significant distances from each other across the three training phases.

The results of the ablation study are presented in Table II. The Meta-manipulation Diffusion Policy significantly outperforms its counterparts that do not use the Meta Vector for additional contextual conditioning or integrate the Current Endpos vector with the current estimate action for fusion, demonstrating the critical roles these components play in boosting the model's effectiveness. Figure 6 illustrates the changes in meta vectors of MetaDP with D.T. across three training phases. The meta vectors maintain significant distances from each other throughout these phases, highlighting the substantial differences between the various task stages and underscoring the necessity of the meta vectors.

Real-World Experiments. We assess the performance of MetaDP on real visual sensory data by training and testing the model in a real-world setup. We conduct our experiments on a tabletop setup featuring a stationary Universal Robots UR3 robotic arm. The scene is captured from a third-person perspective using a statically mounted Intel RealSense D435i RGB-D camera. We acquire color and depth images at a resolution of 640×480 pixels and perform alignment between them. We then crop the central 256×256 region from the color image to serve as the image input for our model. The depth image undergoes the same cropping process, and using the intrinsic parameters of the color camera, we convert the depth data into a point cloud, which serves as the point cloud input for our model. Motivated by the goal of addressing fine manipulation challenges in robotic arms, we design precise insertion tasks to evaluate the model's performance, as illustrated in Figure 7.

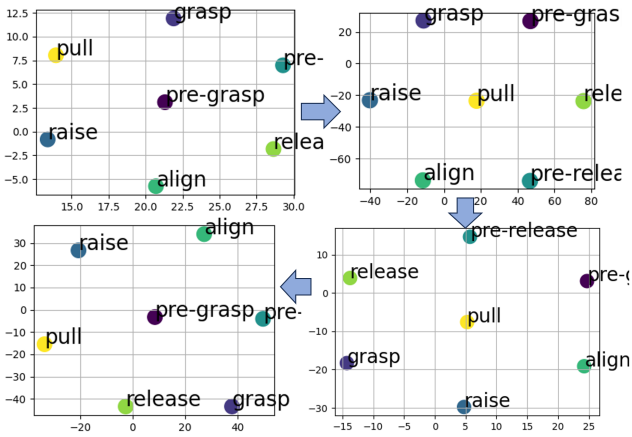

Fig. 6: **2D Visualization of MetaDP with D. T.** The top-left corner represents the meta vector at network initialization. Following the arrows, the subsequent images show the meta vectors after training on six individual tasks, then after training on one additional task, and finally after training on yet another task. All visualizations are performed using t-SNE.

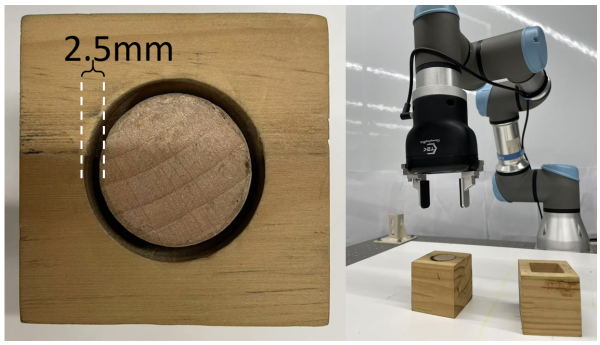

Fig. 7: **Overview of the real-world experiment.** The cylindrical object with a diameter of 3 cm is grasped and inserted into a hole. The allowable tolerance on each side of the hole is 2.5 mm.

Data Collection. We provide real-world datasets for the MetaDP model through human demonstrations. Since our work is based on Keypose, we do not need to collect trajectories during motion. Instead, we predefine five keypoints: pre-grasp, grasp, raise, pre-release, and release. We employ a reverse data collection method where we first secure the cylinder with the gripper and then manually guide the robotic arm to insert it into the corresponding hole. By reversing these keypoints, we collect the corresponding environment states and actions. Finally, we invert the data sequence to complete the data collection process. For a single task, we collected 44 expert demonstrations, with 4 used as validation datasets. We trained the model for 40,000 steps with a batch

size of 20 on an RTX 3090 GPU.

Result. We randomly arrange objects in the scene, and use the insertion method for path planning during the execution of actions to ensure the robotic arm's straight-line motion. We conducted 10 tests on high-precision insertion tasks, and under the condition of using a single camera, the task success rate reached 60%. Under the same conditions, our algorithm demonstrated a higher success rate, indicating a certain level of improvement. For a demonstration, please refer to the following video: https://youtu.be/if0vIpeAeVg

REFERENCES

[1] A. Goyal, J. Xu, Y. Guo, V. Blukis, Y. Chao, and D. Fox. RVT: Robotic view transformer for 3d object manipulation. In *Proc. Conference on Robot Learning*, pages 694–710, 2023.

[2] 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.

[3] M. J. Kim, K.l Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. OpenVLA: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[4] A. Brohan, N. Brown, et al. RT-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[5] T.Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[6] S. Chen, R. Garcia, C. Schmid, and I. Laptev. Polarnet: 3d point clouds for language-guided robotic manipulation. *arXiv preprint arXiv:2309.15596*, 2023.

[7] W. Yuan, A. Murali, A. Mousavian, and D. Fox. M2T2: Multi-task masked transformer for object-centric pick and place. *arXiv preprint arXiv:2311.00926*, 2023.

[8] M. Shridhar, L. Manuelli, and D. Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Proc. Conference on Robot Learning*, pages 785–799, 2023.

[9] H. Huang, O. Howell, X. Zhu, D. Wang, R. Walters, and R. Platt. Fourier transporter: Bi-equivariant robotic manipulation in 3d. *arXiv preprint arXiv:2401.12046*, 2024.

[10] T. Gervet, Z. Xian, N. Gkanatsios, and K. Fragkiadaki. Act3d: Infinite resolution action detection transformer for robotic manipulation. *arXiv preprint arXiv:2306.17817*, 2023.

[11] Z. Xian, N. Gkanatsios, T. Gervet, T.-W. Ke, and K. Fragkiadaki. Chaineddiffuser: Unifying trajectory diffusion and keypose prediction for robotic manipulation. In *Proc. 7th Annual Conference on Robot Learning*, 2023.

[12] Jiayi Chen, Ying He, and F Richard Yu. Tagguidebot: Enhancing robot intelligence with object tags and vlms. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 842–849. IEEE, 2025.

[13] Fan Zhuo, Ying He, Fei Yu, Pengteng Li, Zheyi Zhao, and Xilong Sun. Abm: Attention before manipulation. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24, K. Larson, Ed. International Joint Conferences on Artificial Intelligence Organization*, volume 8, pages 1816–1824, 2024.

[14] A. Brohan, N. Brown, J. Carbajal, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[15] S. Belkhale, T. Ding, T. Xiao, P. Sermanet, Q. Vuong, J. Tompson, Y. Chebotar, D. Dwibedi, and D. Sadigh. RT-H: Action hierarchies using language. *arXiv preprint arXiv:2403.01823*, 2024.

[16] W. Huang, R. Wang, C.and Zhang, Y. Li, J. Wu, and F.-F. Li. Voxposer: Composable 3D value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

[17] H. Huang, F. Lin, Y. Hu, S. Wang, and Y. Gao. Copa: General robotic manipulation through spatial constraints of parts with foundation models. *arXiv preprint arXiv:2403.08248*, 2024.

[18] Zheyi Zhao, Ying He, Fei Yu, Pengteng Li, Fan Zhuo, and Xilong Sun. Llakey: Follow my basic action instructions to your next key state. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9604–9611. IEEE, 2024.

[19] Y. Du, S. Yang, B. Dai, H. Dai, O. Nachum, J. Tenenbaum, D. Schuurmans, and P. Abbeel. Learning universal policies via text-guided video generation. *Advances in Neural Information Processing Systems*, 36, 2024.

[20] T. Yu, T. Xiao, A. Stone, J. Tompson, A. Brohan, S. Wang, J. Singh, C. Tan, J. Peralta, B. Ichter, et al. Scaling robot learning with semantically imagined experience. *arXiv preprint arXiv:2302.11550*, 2023.

[21] C. Lu, P. Ball, Y. W. Teh, and J. Parker-Holder. Synthetic experience replay. *Advances in Neural Information Processing Systems*, 36, 2024.

[22] H.-C. Wang, S.-F. Chen, M.-H. Hsu, C.-M. Lai, and S.-H. Sun. Diffusion model-augmented behavioral cloning. *arXiv preprint arXiv:2302.13335*, 2023.

[23] M. Janner, Y. Du, J. B. Tenenbaum, and S. Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

[24] A. Ajay, Y. Du, A. Gupta, J. Tenenbaum, T. Jaakkola, and P. Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.

[25] X. Li, V. Belagali, J. Shang, and M. S. Ryoo. Crossway diffusion: Improving diffusion-based visuomotor policy via self-supervised learning. In *Proc. IEEE International Conference on Robotics and Automation*, pages 16841–16849, 2024.

[26] S. Gubbi, S. Kolathaya, and B. Amrutur. Imitation learning for high precision peg-in-hole tasks. In *Proc. 6th International Conference on Control, Automation and Robotics*, pages 368–372, 2020.

[27] B. Tang, M. A. Lin, I. Akinola, A. Handa, G. S. Sukhatme, F. Ramos, D. Fox, and Y. Narang. Industreal: Transferring contact-rich assembly tasks from simulation to reality. *arXiv preprint arXiv:2305.17110*, 2023.

[28] M. A. Lee, Y. Zhu, K. Srinivasan, P. Shah, S. Savarese, Li Fei-Fei, A. Garg, and J. Bohg. Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks. In *Proc. 2019 International Conference on Robotics and Automation*, pages 8943–8950, 2019.

[29] J. Xu, S. Kim, T. Chen, A. R. Garcia, P. Agrawal, W. Matusik, and S. Sueda. Efficient tactile simulation with differentiability for robotic manipulation. In *Proc. Conference on Robot Learning*, pages 1488–1498, 2023.

[30] Angela W. Yu and Amiya Nayak. The Internet of humanoids: A survey of technologies, applications, and challenges. *IEEE Internet of Things Journal*, 2026. Online early access.

[31] A. Goyal, V. Blukis, J. Xu, Y. Guo, Y.-W. Chao, and D. Fox. RVT-2: Learning precise manipulation from few demonstrations. *arXiv preprint arXiv:2406.08545*, 2024.

[32] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[33] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li. On the continuity of rotation representations in neural networks. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5745–5753, 2019.

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

[35] F. Richard Yu. *Intropy: A Framework for Modeling Intelligence*. Amazon Digital Services, 2026. Kindle edition.

[36] Y. Ren, H. Zhang, F. R. Yu, et al. Industrial internet of things with large language models (llms): An intelligence-based reinforcement learning approach. *IEEE Trans. Mobile Computing*, 24(5):4136–4152, 2025.

[37] E. Rohmer, S. PN Singh, and M. Freese. V-REP: A versatile and scalable robot simulation framework. In *Proc. 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1321–1326, 2013.

[38] S. James, M. Freese, and A. J. Davison. Pyrep: Bringing v-rep to deep robot learning. *arXiv preprint arXiv:1906.11176*, 2019.

[39] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison. Rlbench: The robot learning benchmark learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.